xassp User's Manual (Advanced Speech Signal Processor under the X Window System)

IPDS, Kiel

Contents

1 Introduction **39**
- 1.1 What is *xassp*? . . . 39
- 1.2 Manual Structure . . . 39
- 1.3 Typographic conventions . . . 40

2 Getting Started with *xassp* **41**
- 2.1 Starting *xassp* . . . 41
- 2.2 Opening Files . . . 42
- 2.3 Segmental Labelling . . . 43
  - 2.3.1 Fetching a Label . . . 43
  - 2.3.2 Playing the Speech Signal . . . 46
  - 2.3.3 Using the Label List . . . 46
  - 2.3.4 Fetching and Modifying a Label . . . 47
  - 2.3.5 Modifying a Label after Moving it . . . 47
  - 2.3.6 Inserting a New Label . . . 47
  - 2.3.7 Label Syntax Checks . . . 47
  - 2.3.8 Deleting an Inserted Label . . . 48
  - 2.3.9 Editing an Inserted Label . . . 48
  - 2.3.10 Moving a Label . . . 48
  - 2.3.11 Undoing Recent Changes . . . 48
  - 2.3.12 Jumping to a Specified Label . . . 48
  - 2.3.13 Saving the Labels in a File . . . 49
- 2.4 Prosodic Labelling . . . 49
  - 2.4.1 Inserting a Prosodic Label . . . 51
  - 2.4.2 Editing a Label . . . 51
  - 2.4.3 Deleting a Label . . . 51
  - 2.4.4 Undoing Recent Changes . . . 51
- 2.5 Analysing Speech Signals . . . 51

3 Using *xassp* **53**
- 3.1 User Levels . . . 53
- 3.2 *xassp* Command Line Options . . . 53
  - 3.2.1 Examples . . . 54

3.3 The *xassp* Main Dialog 55
3.3.1 The File Selection Box 55
3.3.2 File Type Buttons 56
3.3.3 Options for loading speech signal data in other formats 56
3.3.4 Align and Link 57
3.3.5 Data Logging 57
3.3.6 Configuration Buttons 57
3.3.7 Additional Buttons 57
3.4 Common *xassp* Window properties 57
3.4.1 Elements of *xassp* Windows 58
3.4.2 Keyboard and Mouse 61
3.4.3 Pop-up Menus 61
3.4.4 Window Linking 63
3.4.5 Data Logging 63
3.5 Specific *xassp* Window Properties 65
3.5.1 The Speech Signal Window 65
3.5.2 The Sonagram Window 67
3.5.3 The Section Window 69
3.5.4 The Label Window 71
3.5.5 The Fundamental Frequency Window 73
3.5.6 The Energy Window 75
3.6 File Handling 76
3.6.1 Speech Signal Files 77
3.6.2 Fundamental Frequency 78
3.6.3 Energy 78
3.6.4 Labels 78
3.7 Printing 81
3.7.1 Modifying Page Options 81
3.7.2 Selecting Windows 82
3.7.3 Choosing a Print Configuration 82
3.7.4 Modifying Window Parameters 83

4 Configuring *xassp* 85
4.1 X Resources 85
4.2 User Configuration Files 87
4.2.1 The Main Configuration File 88
4.2.2 The Font Configuration File 89
4.2.3 The Label Configuration File 90
4.3 Administrative Configuration Files 90
4.3.1 The *xassp* Users File 90

5 Analyses **93**
5.1 Energy Analysis 93
5.2 F_0 or Pitch Analysis 94
5.3 Formant Analysis 95
5.4 Spectral Analysis 96

6 Definition of Signal Processing Terms **97**

A Key and Mouse Bindings **105**

B X Resources **109**
B.1 Resources for the *Core* widget 109
B.2 Resources for the *XmPrimitive* widget 110
B.3 Resources for the *XspMain* widget 111
B.4 Resources for the *XspData* widget 112
B.5 Resources for the *XspWave* widget 112
B.6 Resources for the *XspFzero* widget 112
B.7 Resources for the *XspLabels* widget 112
B.8 Resources for the *XspSonag* widget 113
B.9 Resources for the *XspSection* widget 113
B.10 Resources for the *XspEnerg* widget 114
B.11 Resources for the *XspHscale* widget 114
B.12 Resources for the *XspVscale* widget 115

Foreword

As early as 1979, after acquiring its first computing system for phonetic research (a Data General Eclipse S/230), the Institute of Phonetics at Kiel University started developing its own custom built speech analysis and synthesis software. The initial result was the program package SSP — Speech Signal Processor by Kurt Schäfer-Vincent (see Barry et al. 1982), which allowed electronic splicing of signals, analyis of acoustic parameters (especially with a very powerful pitch algorithm, see Schäfer-Vincent 1982,1983), parameter manipulation and LPC synthesis. With the replacement of the Eclipse computer by a net of phonetic work stations this package was transferred to an OS9/68k environment, expanded and made more flexible by Michel Scheffers. The outcome was ASSP — Advanced Speech Signal Processor (see Scheffers and Thon 1991). Eventually, Michel Scheffers transported ASSP to a Unix framework — Apollo computers and PCs running on Linux.

In 1990 the Department of Speech Communication and Music Acoustics of KTH Stockholm gave IPDS their MIX program software, which was intended to be used for labelling speech files on Apollo computers (see Carlson and Granström 1986), "mixing" signal files and prototype label files. The latter were automatically generated from orthographic text within the Rulsys/Infovox text-to-speech system, which IPDS Kiel adapted for German, including spontaneous speech (see Kohler et al. 1995). The MIX software was adjusted to the German segmentation and labelling conventions and made more flexible and user friendly by a student of computer science at IPDS, Frank Bartels. It was then used very extensively in the BMFT/BMBF funded ASL and VERBMOBIL projects. The German speech data processed with the help of MIX have been distributed on three CD-ROMs (The *Kiel Corpus of Read/Spontaneous Speech*).

As the Apollo hard- and software ceased to be supported it became necessary to think about a future platform for segmentation and labelling of German speech data of various speaking styles, all the more so since more and more corpora labelled interactively by hand were required but the number of our Apollo work stations was extremely small and thus could not meet the demand. Matthias Pätzold and Adrian Simpson worked out a general concept for combining the ASSP speech processing algorithms with the MIX framework of speech signal segmentation and labelling in a comprehensive speech processing software package portable to any Unix environment under the X Window System. This was first realised for prosodic labelling within the Kiel PROLAB notation (see Kohler et al. 1995) and then extended to segmental la-

belling. A student of physics at IPDS, Tobias Rettstadt, programmed the *X Windows* implementation, Michel Scheffers incorporated the ASSP algorithms in this frame. The final product is *xassp*.

This User's Manual of *xassp* provides a guide to the application of the program package for anyone that wishes to segment, label and analyse speech data in any language. The handbook was compiled jointly by Tobias Rettstadt, Michel Scheffers, Claudia Rehor and myself.

Kiel, November 1997 *Klaus J. Kohler*

Chapter 1

Introduction

1.1 What is *xassp*?

xassp refers to the IPDS **A**dvanced **S**peech **S**ignal **P**rocessing tool, which operates under the ***X** Window System*.

xassp can be used for a wide range of purposes. The most important ones are the analysis and display of speech signals including fundamental frequency (F_0), energy, and spectrum (sonagram, section). Another important aspect is the possibility of assigning segmental and prosodic labels to distinct points in time.

1.2 Manual Structure

This chapter (Chapter 1) contains a short description of *xassp* and this manual.

Chapter 2 shows how to use *xassp* for the most common tasks. It is intended to be a tutorial-like introduction to *xassp*.

Chapter 3 provides a complete description of all features of *xassp*. It explains the data types that *xassp* can handle and also shows different ways to create, load, display and manipulate them.

Chapter 4 deals with the numerous configuration possibilities of *xassp*. These include *X resources* as well as *xassp*'s own configuration files.

Chapter 5 explains the methods for analysing speech signals that *xassp* is capable of.

Chapter 6 contains definitions of the most important signal processing terms that are used throughout the manual.

Appendix A provides an overview of all key and mouse bindings that are available in *xassp*.

Appendix B contains a reference list of *X resources* that have an influence on the behaviour of *xassp*.

1.3 Typographic conventions

When referring to keys or mouse buttons, the following conventions are used in this manual:

A single keystroke always appears as a single letter set in `typewriter` font, e.g. `g` means hitting the key labelled G. Special keys such as the *Return* key are written in **bold face**, e.g. **Return**. The mouse buttons are also treated as special keys: **LM** (left mouse button), **MM** (middle mouse button), **RM** (right mouse button).

Modifier keys (keys that must be held down while pressing another key) are written as <**Shift**>, <**Ctrl**>, <**Alt**>. <**Ctrl**>**RM** means pressing the right mouse button while holding down the Ctrl key.

Chapter 2

Getting Started with *xassp*

This chapter will help you to get to know *xassp* and its basic principles. It describes how to perform the most common tasks, while Chapter 3 provides a complete description of the features of *xassp*.

How *xassp* configuration files and resources may be changed will be set out in Chapter 4.

2.1 Starting *xassp*

To start *xassp* simply type

```
xassp
```

on the command line and press **Return**. If you want to do prosodic labelling, type

```
xassp -u10
```

followed by **Return** to start *xassp*. If you want to do segmental labelling, you have to invoke *xassp* by typing

```
xassp -u20
```

again followed by **Return**.

The numbers in the last two commands are the user levels that are required to do prosodic (user level 10) or segmental (user level 20) labelling.

Please refer to Section 3.1 for more information on user levels and to Section 3.2 for a complete description of *xassp* command line options.

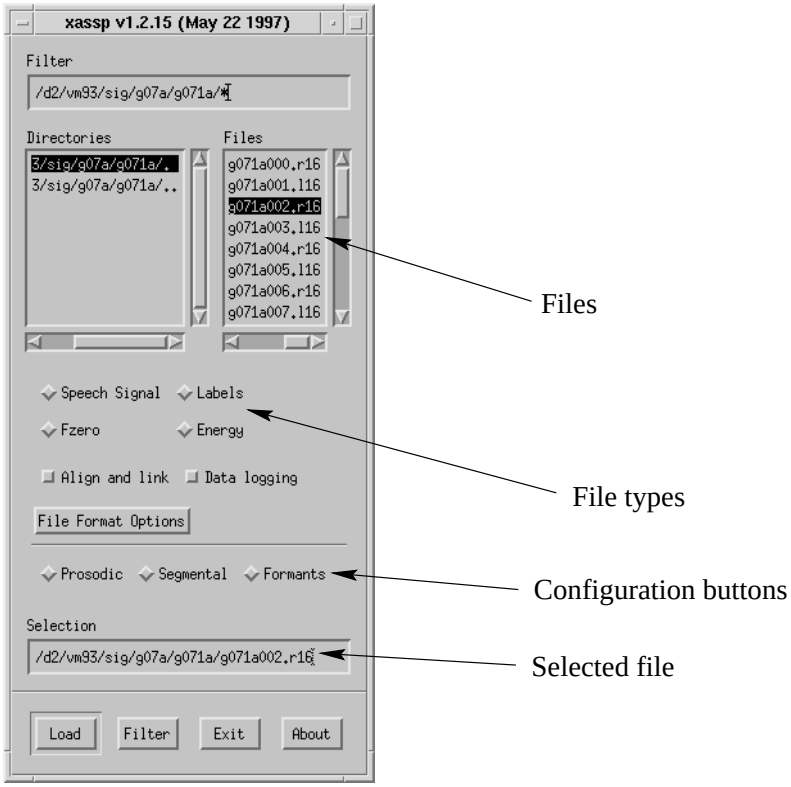

Figure 2.1: *xassp* Main Dialog

2.2 Opening Files

After starting *xassp* the *xassp* main dialog appears on the screen (see Figure 2.1). Now files are selected for display in windows. There are several ways to do this. The first step is choosing the directory that contains the file of interest. This can be done either by typing the path in the text field labelled *Filter* or by selecting a directory from the *Directories* list. When typing the path remember to add a slash (/) at the end. To tell *xassp* to display the contents of the chosen directory you can either press **Return** in the *Filter* text field or click on the *Filter* button or double-click on the directory in the *Directories* list.

The next step is choosing the file to be loaded. You can either select the file from the *Files* list or type the file name in the *Selection* field.

Now you can actually load the file by clicking on the *Load* button. If *xassp* refuses to load the file, you have to specify the file type by selecting one of the buttons labelled *Speech Signal*, *Fzero*, *Energy*, and *Labels*. Then try to open the file again by pressing the *Load* button. If *xassp* still displays an error message, the file format is not recognised or you selected the wrong file type.

If the file you chose for display consists of raw speech signal data or data in an unsupported format, you can specify further options by clicking on the *File Format*

Options button. These are explained in Section 3.3.3.

If the *Align and link* button is selected when loading a file, all windows are aligned and linked after the load operation is complete. For more information on links, see Section 3.4.4. Selecting this button only makes sense if you have already opened one or more *xassp* windows.

Instead of opening the files one after another you can use configurations. To do this you select one of the files for loading, select one of the configuration buttons (they are located directly above the *Selection* field, see Figure 2.1) and press the *Load* button. The configuration *Prosodic*, e.g., would open a speech signal window, a fundamental frequency window and a label window. The *Segmental* configuration opens a speech signal window, a sonagram window and a label window. In both cases a speech signal file with the suffix `l16`, `r16`, `shh`, `wav`, `syn` or `raw` and a label file with the suffix `mix` must be present. The fundamental frequency and the sonagram are computed from the speech signal.

2.3 Segmental Labelling

Segmental labelling in *xassp* has been especially tailored to meet the needs of the signal annotation carried out at *ipds* (Kohler, Pätzold, and Simpson 1995). Before you can begin with segmental labelling you must load a set of files including the speech signal file and the label file. It is also helpful to use a sonagram that is computed from the speech signal. You can load the files, do the analysis, align and link the window by selecting the configuration *Segmental* as it is described in Section 2.2. If you use this configuration, the windows that appear on the screen should look similar to those that are shown in Figure 2.2.

When doing segmental labelling labels are set at segment boundaries. The labels and the order in which they are to be placed are given (in the following the term *label stack* will be used for the given labels). The label order cannot be changed and labels that are taken from the label stack cannot be deleted. New labels can only be inserted if they are also marked as inserted with a `-` (minus) after the label prefix.

2.3.1 Fetching a Label

To place the next label from the label stack, which is shown in the upper right corner of the label window, at a certain point in time, you first have to set the temporary marker onto this point (see Figure 2.3). This can be achieved by placing the cursor at this point and then pressing <**Shift**>**LM**. Then you have to move the cursor between the temporary marker and the end of the window. Pressing <**Shift**>**MM** moves the next label from the label stack to the point in time at which the temporary marker is set. This procedure of setting the temporary marker and fetching a label from the label stack is repeated until there are no more labels left on the stack. In this case the small box in the upper right corner that displays the next label disappears.

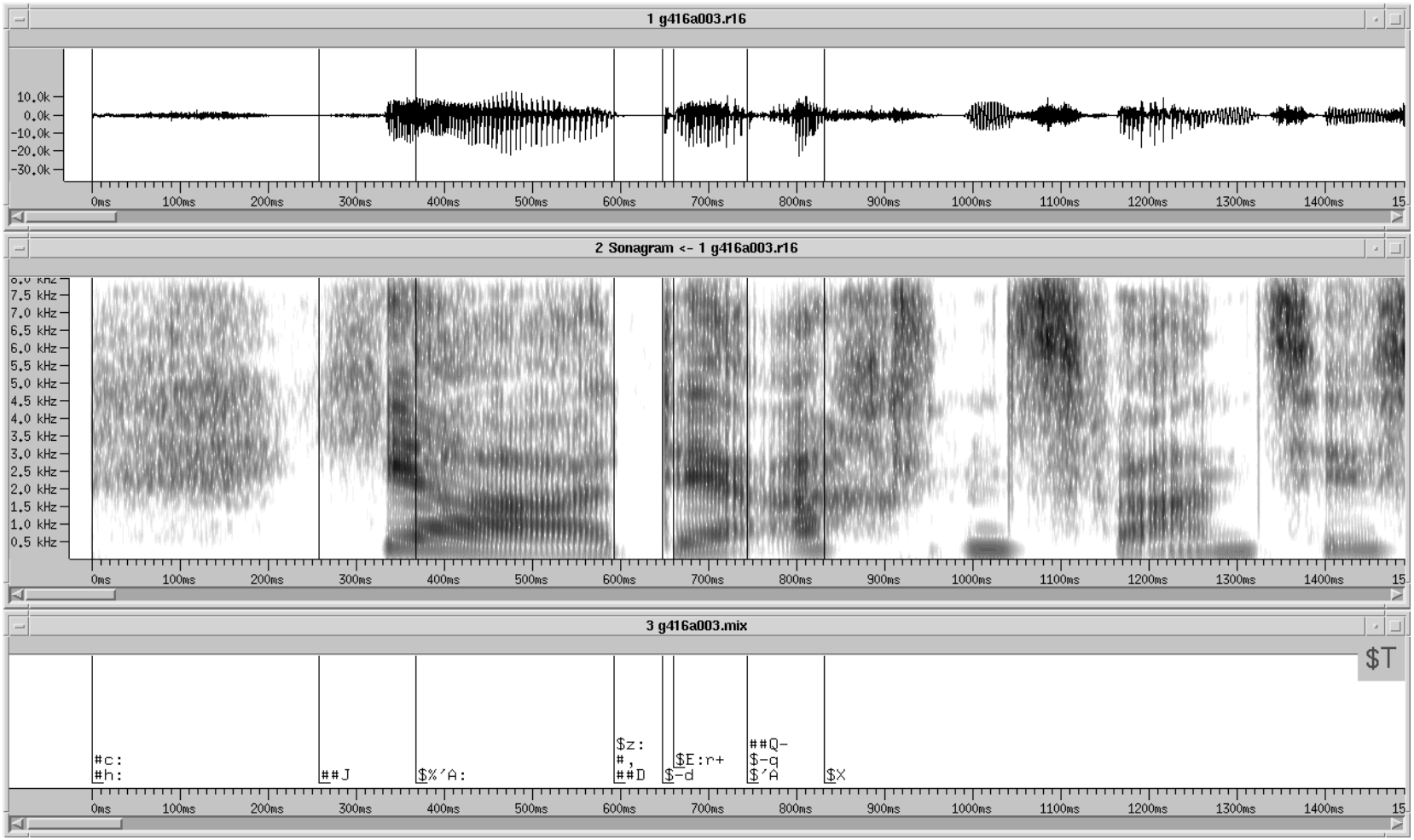

Figure 2.2: Window configuration for segmental labelling. Upper window: speech wave, middle window: sonagram, lower window: labels (this file has been partly segmented)

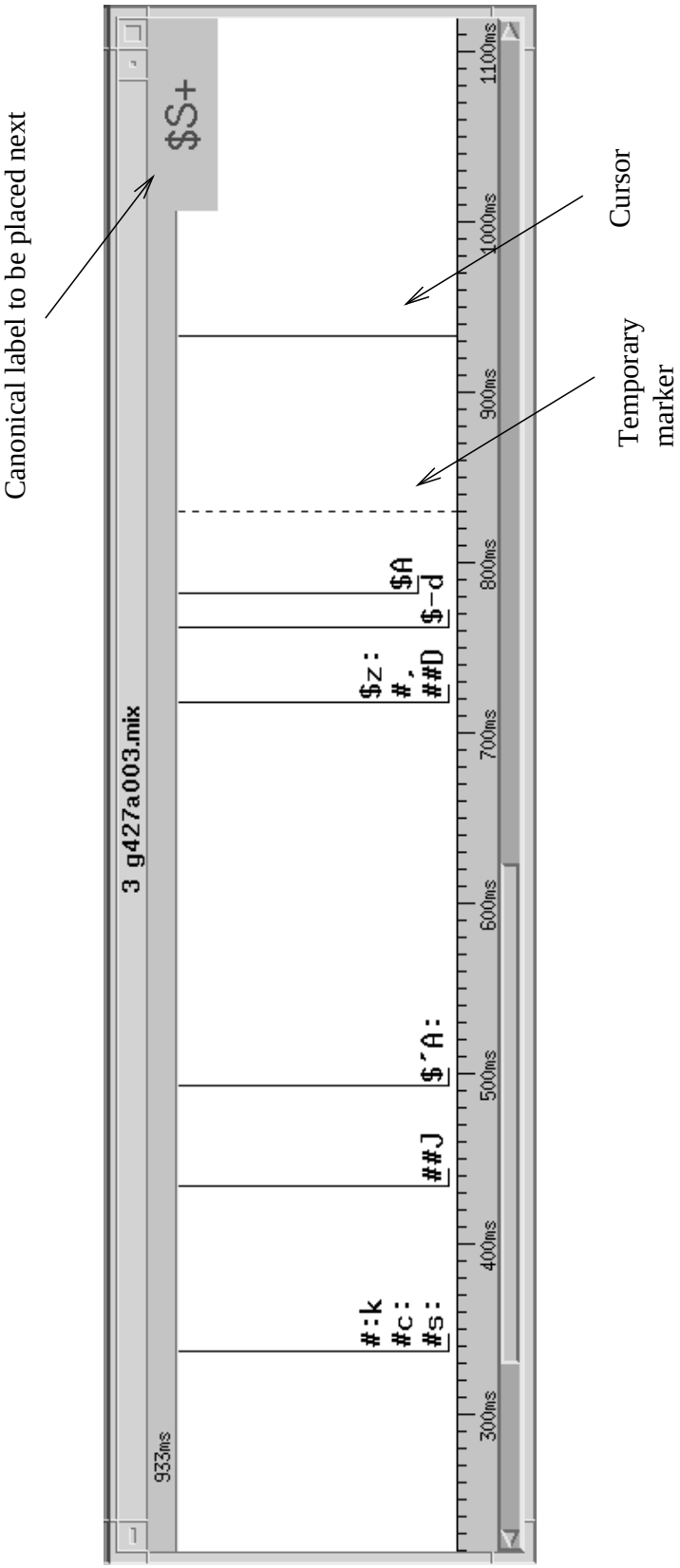

Figure 2.3: Label placement

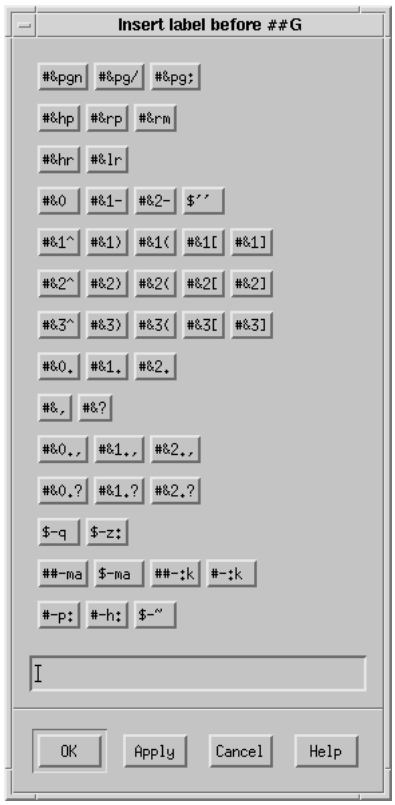

Figure 2.4: Edit Dialog Box in the Label Window. The place where the label is to be inserted is indicated in the dialog title, in this case the label will be inserted before the label ##G.

2.3.2 Playing the Speech Signal

It is difficult to determine segment boundaries without hearing the relevant stretch of the speech signal. You can repeatedly play the signal from the temporary marker to the cursor with **LM** and shift the temporary marker (by pressing <**Shift**>**LM**) until you think that it is set on the right point. You can also play other parts of the speech signal (see Tables A.1 and A.2).

2.3.3 Using the Label List

xassp is able to list the labels of the current file in a separate window. You can open this window by pressing <**Alt**>**RM** and selecting *Show as Text* from the pop-up menu that appears. The label list consists of two columns. In the first column the time in milliseconds is displayed. The second column contains the labels. The labels on the label stack normally have a time of 100,000 milliseconds. The latter can only be seen in the label list, they are not displayed in the normal label window. If you make changes to the labels in the label window, the label list is updated.

2.3.4 Fetching and Modifying a Label

Since the labels on the label stack do not always match the utterance that is to be labelled there are several possibilities to modify them. If you press <**Shift**>**RM** instead of <**Shift**>**MM** to fetch the label from the label stack, you can edit the label before it is moved. In this case an *Edit Label* dialog box is displayed (see Figure 2.4). In this dialog box you can edit the label in the text field that is located directly above the three buttons labelled *OK*, *Cancel* and *Help*. Pressing the *OK* button then moves the edited label onto the temporary marker, and pressing the *Cancel* button neither changes nor moves the label.

The most frequent changes that are made to labels taken from the stack are appending a minus sign to mark the label as deleted and inserting a percent sign to mark an uncertain segment boundary. These frequent modifications have been assigned directly to mouse buttons. You can fetch a label and append a minus sign by pressing <**Alt**>**LM** and fetch the label and insert a percent sign by pressing <**Ctrl**>**LM**.

2.3.5 Modifying a Label after Moving it

If you moved the label and then realise that you forgot to modify it, you can still edit it by moving the mouse pointer onto the label (the label is highlighted), pressing <**Ctrl**>**RM** and selecting *Edit* from the pop-up menu that appears. The *Edit Label* dialog box is exactly the same as the one described above except that the label is not moved if you press the *OK* button.

Note that you are only allowed to make changes that are absolutely necessary, e.g., changing the label prefix from ## (word boundary) to $ (word-internal).

2.3.6 Inserting a New Label

If there is no label on the stack that matches the stretch of the speech signal that has to be labelled next, you can insert a new label at the temporary marker with <**Ctrl**>**MM**. This label must be marked as inserted by inserting a minus sign after the label prefix. If you forget to do this, *xassp* refuses to insert the label and shows a warning message.

2.3.7 Label Syntax Checks

The check that is done when inserting a label is only one of a number of label syntax checks that *xassp* performs whenever a label is edited, inserted or deleted. So, for instance, you are not allowed to insert invalid labels or to edit labels so that they become invalid. A more detailed description of these checks can be found in Section 3.5.4.

2.3.8 Deleting an Inserted Label

If you accidentally inserted a label and want to delete it, you have to move the mouse pointer onto the label (the label is highlighted), press <**Ctrl**>**RM** and select *Delete* from the pop-up menu that appears. You will not be able to delete any labels that were fetched from the label stack, since a check is performed before the label is deleted. If you try to do this anyway, a warning message is displayed and the label is not deleted.

2.3.9 Editing an Inserted Label

xassp refuses any attempt to edit an inserted label (at least if your user level is less than 30). To edit an inserted label you therefore have to delete and re-insert it at the same point in time as described above.

2.3.10 Moving a Label

When you discover that a label was not set onto the right point you will want to move it onto a different one. This action is very similar to fetching a label from the label stack. You first have to set the temporary marker to the point in time onto which you would like to move the label. Then place the cursor between the temporary marker and the label that is to be moved. Note that only a label directly to the left or directly to the right of the temporary marker can be moved. The label order cannot be changed. By pressing <**Shift**>**MM** you can now move the label onto the temporary marker. You can also use the short-cuts that are described above to edit the label before moving it or to modify it by inserting a percent sign, and so on.

2.3.11 Undoing Recent Changes

xassp remembers every change that you make (inserting, deleting, moving and editing labels). By pressing <**Ctrl**>**RM** and then selecting *Undo* from the pop-up menu that appears you can undo the last change. Repeating this action undoes the last but one change, and so on. You can continue undoing until the first change that you made is reversed.

2.3.12 Jumping to a Specified Label

There are several nice features that make segmental labelling with *xassp* more user-friendly. If you want to label a file that has already been partly labelled, you can jump to the last label by pressing <**Ctrl**>`l` after loading the file.

<**Ctrl**>`s` makes the *Go to* dialog box appear on the screen. In its *Label* text field you can type a label you wish to jump to. After pressing the *Go* button, the label you entered is searched for, starting from the current position of the beginning of the window and the first occurrence of the label. If the label has been found, the label

window is scrolled so that the label is displayed in the centre of the window, if the label has not been found, a warning message is displayed. Now you can jump to the next (previous) occurrence of the same label by pressing <**Ctrl**>`f` (<**Ctrl**>`b`). If there is no next (previous) occurrence, a warning message is displayed.

2.3.13 Saving the Labels in a File

If you press the *Exit* button in the *xassp* main dialog box or select *Close* from the label window menu or *Close All* from the window menu of any open *xassp* window, you are prompted whether you want to save the data in case you made changes to them. Press *Save* to save the data and exit, press *Do not save* to exit *xassp* without saving any changes. Pressing *Cancel* neither saves the data nor exits *xassp*.

To save the data without exiting *xassp* press <**Alt**>**RM** and select *Save* from the pop-up menu that appears. If an error during the writing of the data, an error message is displayed on the screen.

You can save the labels under a different file name by pressing <**Alt**>**RM** and selecting *Save As* from the pop-up menu. You can then type the new file name in the *Selection* text field or select a file from the *Files* list. Pressing the *OK* button saves the data to the specified file. If the file you selected already exists, you are prompted whether the file should be overwritten. If the data were successfully written, a short message appears.

2.4 Prosodic Labelling

The first step for prosodic labelling is to load a set of files including a speech signal file and a label file. In most cases you will also need the fundamental frequency of the speech signal. If you select the configuration *Prosodic* as described in Section 2.2, the speech signal and label files are loaded, a fundamental frequency analysis of the speech signal is done and the corresponding windows are opened as shown in Figure 2.5).

As with segmental labelling, *xassp* has been especially tailored to facilitate the type of prosodic annotation carried out at *ipds*. Among other things this assumes a prior segmental annotation. The segmental labels are then used as the temporal points at which prosodic labels can be placed. Running *xassp* at the appropriate user level (see section 3.1) ensures that prosodic labels can be inserted and modified without affecting segmental information. Of course, as with segmental labelling all checking and restrictions on labels can be overridden if *xassp* is run using a higher user level (`-u 30`).

When doing prosodic labelling new labels are only set at points in time at which the segmental labels are already placed. Therefore there is no need for a temporary marker. The prosodic labels are not fetched from a stack. You can insert any prosodic label at any point in time at which a segmental label is set.

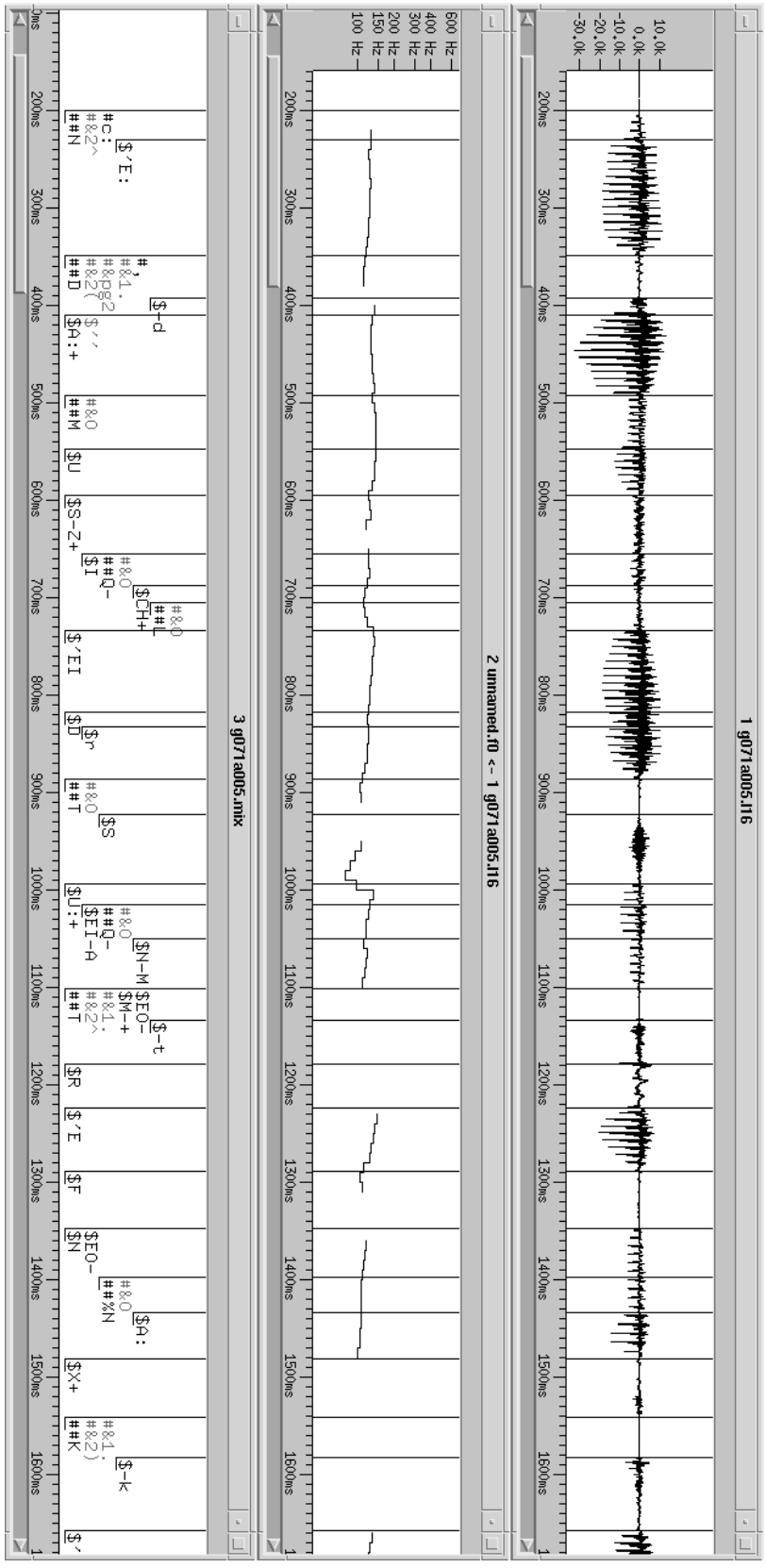

Figure 2.5: Window configuration for prosodic labelling. Upper window: speech wave, middle window: fundamental frequency, lower window: labels

2.4.1 Inserting a Prosodic Label

You can insert a prosodic label before or after a given label. To do this you move the mouse pointer onto the label before or after which the new label is to be inserted (the label is highlighted), press <**Ctrl**>**RM** and select *Insert before* or *Insert after* in the pop-up menu that appears. In the *Insert Label* dialog box, which is shown in Figure 2.4, you can either type the label in the text field that is located directly above the row of buttons and then press **Return** to insert the label, or press one of the label buttons provided in the dialog box to insert the label it shows. The latter method is intended to be used for the most frequent labels. How the labels that are shown as buttons in the *Insert Label* dialog box can be changed is described in Section 4.2.3.

Note that inserted prosodic labels, unlike segmental labels, do not require a - (minus) after the prefix and can be edited.

2.4.2 Editing a Label

You can edit an inserted label by moving the mouse pointer onto the label (the label is highlighted), pressing <**Ctrl**>**RM** and selecting *Edit* from the pop-up menu that appears. The *Edit Label* dialog box is the same as the *Insert Label* dialog box. You can either edit the label in the text field, or press one of the buttons. If one of the buttons is pressed, the label is changed to the one that is displayed on the button.

It is also possible to edit segmental labels. To do this you take the same actions as for editing prosodic labels. Note that the changes you can make to segmental labels are restricted.

2.4.3 Deleting a Label

If you accidentally inserted a prosodic label, you can delete it by moving the mouse pointer onto the label (the label is highlighted), pressing <**Ctrl**>**RM** and then selecting *Delete* from the pop-up menu that appears. You are asked whether you really want to delete the label. If you press the *OK* button, the label is deleted. If you press the *Cancel* button, nothing happens.

2.4.4 Undoing Recent Changes

By pressing <**Ctrl**>**RM** you can undo the last change you made. You can repeat this action until the first change is reversed.

2.5 Analysing Speech Signals

The analyses that *xassp* is capable of are

- fundamental frequency,

- energy,
- sonagram,
- section.

Refer to Chapter 5 for more information on these different analysis methods.

There are two different ways to perform an analysis of a speech signal. First, you can use the window menu item *Analysis* in the speech signal window. The second possibility is the definition of analyses in a configuration. The latter method is described in Section 4.2.1, while the former will be described in this section.

The first step in doing an analysis is to open the speech signal that you want to analyse as described in Section 2.2. Then, in the speech signal window press <**Alt**>**RM** and select *Analysis* from the pop-up menu that appears. This opens the *Analysis* dialog box, in which you can select the analysis to be performed. Click on one or more of the buttons labelled *Fzero*, *Energy*, *Labels*, *Sonagram* and *Section*. If you select the *Align and Link* button, all *xassp* windows are aligned and linked after the analysis is done (see Sections 3.4.4 and 3.5.1 for a description of links and window alignment). If you marked a region of the speech signal by placing brackets (with **F5**, **F6**), you can choose whether to analyse the whole speech signal (select the *All* button) or only the marked region (select the *Region* button).

Now you can select the *OK* button to perform the analysis. The outcome of each analysis is presented in a separate window. If you select the *Cancel* button, no analysis is performed and no new windows are opened.

Note that you cannot modify any analysis parameters before the analysis is finished. Instead, you have to press <**Alt**>**RM** in the window that contains the analysis whose parameters you would like to change and select *Re-analyse* from the pop-up menu that appears. The *Options* dialog box appears on the screen. Here you can modify the analysis parameters. After selecting the *OK* button, the speech signal is reanalysed using the modified analysis parameters.

Descriptions of the *Options* dialog boxes are found in Sections 3.5.2 (sonagram), 3.5.3 (section), 3.5.5 (fundamental frequency) and 3.5.6 (energy). For more information on the different analysis methods and their parameters, see Chapter 5.

You may have noticed that there is one button in the *Analysis* dialog box that has so far not been explained. It is the *Labels* button, which, if it is selected and you press *OK*, simply opens an empty label window.

Since the analysis parameters can only be changed after the analysis has been done with default parameters, you can modify the default parameters. This can be achieved by editing the *X* resources used by *xassp*. For more information on *X* resources, see Section 4.1 and appendix B.

Chapter 3

Using *xassp*

3.1 User Levels

User levels restrict the use of *xassp* features excluding those that are not required for the job at hand. In this way modifications of speech signals can, for instance, be prevented when only label editing is required. The user levels needed to perform certain actions in *xassp* have been chosen arbitrarily; they will become configurable in the future.

At user level 0 all actions that involve modifications of any kind of data are blocked. If you choose a user level of 10 or greater, you can edit and delete labels, but at level 10 you may only insert new ones at a point in time if other labels are already set at the same point, and you are not allowed to move labels to a different point in time. This user level is intended for prosodic labelling. To be able to insert new labels and to move old ones, which is needed for segmental labelling, you need a user level of at least 20. With a user level of 30 or more you are even allowed to edit speech signals and save them. Furthermore no label syntax checks are performed at a user level of 30 or more. See Section 3.5.4 for more information on label syntax checks.

User level	Action
0	no editing allowed
10	prosodic labelling
20	segmental labelling
30	editing speech signals

Table 3.1: User Levels

It is important that you always choose the *lowest* user level that allows you to do your work. Table 3.1 displays a short overview helping you in the choice of the appropriate user level.

3.2 *xassp* Command Line Options

xassp can be invoked from the command line (shell) with the following syntax:

```
xassp [-c|--conf <dir>] [-u|--userlevel <user level>]
      [-h|--help]
```

Brackets enclose optional parameters; options that are separated by a vertical bar are equivalent.

With the option `-c` you can specify the local directory where *xassp* searches for its configuration files. The default directory is the subdirectory `.xassp` of the user's home directory.

The option `-u` lets you specify a user level that *xassp* should use during the session. The default user level is taken from the file `/etc/xassp_users`, in which the maximum user level you are allowed is also specified. For more information on this file, see Section 4.3.1.

When the option `-h` is given, *xassp* displays a short description of the command line options and exits. In this case any additional options are ignored.

If you give an option on the command line that is not known to *xassp*, it displays a short message (`illegal option`) and ignores the option. If you forget to specify an argument to the `-c` and `-u` options, *xassp* prints a short message (`option requires an argument`) and ignores the option.

3.2.1 Examples

If you want *xassp* to show you a short help message describing the options, you simply type

```
xassp -h
```

or

```
xassp --help
```

If you want to start *xassp* with user level 20, e.g. to do segmental labelling, you need to type

```
xassp -u 20
```

or

```
xassp --userlevel 20
```

Since all options that are shown above are enclosed in brackets and therefore are optional, you can simply type

```
xassp
```

to invoke *xassp*. The application is then started using the default user level and the default local configuration directory.

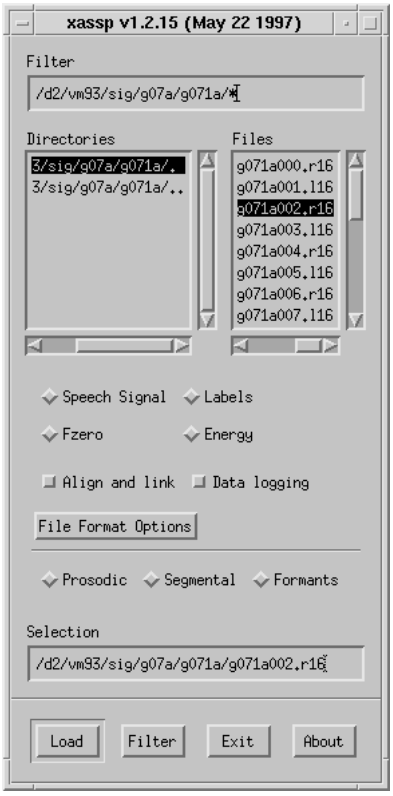

Figure 3.1: *xassp* Main Dialog

3.3 The *xassp* Main Dialog

The *xassp* main dialog (see Figure 3.1) contains

- the file selection box (*Filter* and *Selection* text fields, *Directories* and *Files* lists)
- file type buttons (*Speech signal*, *Fzero*, *Energy* and *Labels*)
- the *File Format Options* button and
- several configuration buttons (for example *Prosodic* and *Segmental*),
- a row of buttons (*Load*, *Filter*, *Exit* and *Help*).

These items will be described in the following sections.

3.3.1 The File Selection Box

The file selection box lets you choose the directory and the actual file that you want to load.

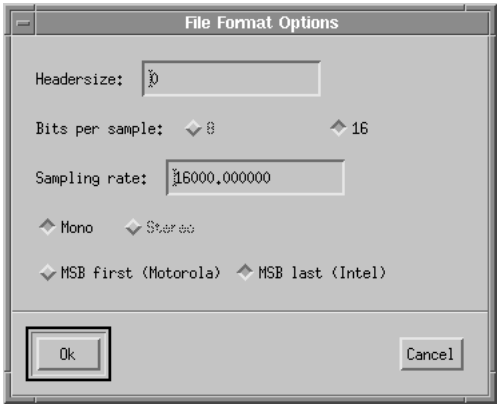

Figure 3.2: File Format Options

To select the directory you use the *Filter* text field, the *Directories* list and the *Filter* button. You can either edit the directory that is given in the *Filter* text field and press **Return** or select the directory in the *Directories* list and then press the *Filter* button. Instead of pressing the *Filter* button it is also possible to double-click on the directory in the *Directories* list.

Now that you are in the right directory you can simply select the appropriate file from the *Files* list.

If you know the complete path to the file that you are interested in you can skip the whole procedure described above and just enter the path including the file name in the *Selection* text field.

Once you have selected a file you can press the *Load* button to load the file and display its contents in a separate window.

3.3.2 File Type Buttons

If *xassp* does not recognise the suffix of the file you selected, you have to specify the file type by selecting one of the buttons labelled *Speech signal*, *Fzero*, *Energy* and *Labels* before pressing the *Load* button. *xassp* then uses the type information you have provided when loading the file.

3.3.3 Options for loading speech signal data in other formats

By pressing the *File Format Options* button in the *xassp* main dialog, the dialog box that is displayed in Figure 3.2 appears. Here you can specify the header size (bytes to skip), the bits per sample (8 or 16), the sampling rate, whether the file is mono or stereo, and the byte order. These options are used, if you load a file that contains raw speech signal data or another format not supported by *xassp* .

3.3.4 Align and Link

Below the file type buttons you find the *Align and Link* button. If this button is set, all *xassp* data windows are aligned and linked after loading some files. See Section 3.4.4 for information on window links.

3.3.5 Data Logging

By clicking the *Data logging* button, which is located next to the *Align and link* button, you can toggle data logging. This feature is explained in Section 3.4.5.

3.3.6 Configuration Buttons

To make it easier for you to open several files at once *xassp* provides configuration buttons, which you find directly above the *Selection* field.

xassp allows the definition of configurations that consist of a set of data types and associated suffixes that describe the files to be loaded. These configurations must be defined in the *xassp* configuration file (see Section 4.2.1).

If you want to open files using a pre-defined configuration, you just start with selecting a single file as described in Section 2.2. *xassp* then determines the files to be opened by substituting the suffix of the selected file by those that are given in the configuration.

It is also possible to specify analyses in the configuration. These are then performed instead of loading a file.

All files that are opened in this way are aligned and linked (see Section 3.4.4 for more information on window linking).

3.3.7 Additional Buttons

There are two buttons that have not been mentioned yet.

The *Exit* button exits *xassp* . If any open *xassp* window contains data that have been modified, you are prompted whether to save them first before exiting.

The *About* button gives information about *xassp*.

3.4 Common *xassp* Window properties

xassp is able to handle the following different data types:

- Speech signal
- Fundamental frequency
- Energy

Data type	Horizontal axis	Vertical axis
Wave	Time	Sample values
Fzero	Time	Frequency
Energy	Time	Signal strength
Labels	Time	n/a
Sonagram	Time	Frequency
Section	Frequency	Spectrum level

Table 3.2: *xassp* Data Types

- Labels
- Sonagram
- Section

To display each of these data types *xassp* uses windows with different properties. The main differences between the data windows are the dimensions represented by the horizontal and vertical axes, which are shown in Table 3.2.

Nevertheless, *xassp* windows have a lot in common, which will be described in this section.

3.4.1 Elements of *xassp* Windows

As you can see in Figure 3.3, *xassp* windows consist of the following elements:

Window title: In the title area *xassp* displays the window number and the name of the file the data were loaded from. If this window contains analysed data, the name of the window the data were derived from is shown as well.

Data area: In this part of the window the data are plotted.

Cursor: The cursor is a vertical bar that is drawn into the data area and always has the same horizontal position as the mouse pointer.

Status line: The status line provides information about the positions of the cursor and the brackets. For displaying these values the same unit as for the horizontal scale is used. The cursor position in the speech signal window is displayed in milliseconds because the horizontal axis is a time scale. In the section window the cursor position is displayed in Hertz.

In addition, the status line also shows the data value at the cursor position using the same unit as the vertical scale. The data value in the energy window, e.g., is displayed in dB.

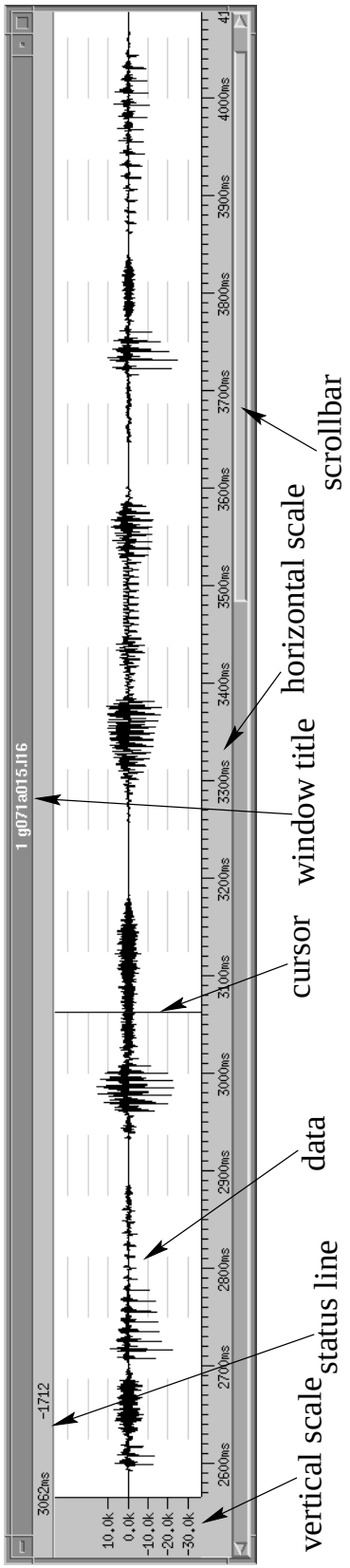

Figure 3.3: *xassp* window containing a speech signal

Horizontal and vertical scales: The scales allow you to associate the data plotted in the data area with values. In the energy window, for example, you can easily find the energy value at a certain point in time by locating this point on the horizontal (time-)scale and then reading the value on the vertical scale. A better way would be to move the cursor to the point in time you are interested in and then read the data value that is shown in the status line.

Scroll bar: Since the width of the data area is limited you cannot always see all data at the same time. The data area therefore only shows a part of the data. The scroll bar lets you choose which part to display.

Pop-up menus: Pressing <**Alt**>**RM** makes a pop-up menu (window menu) appear on the screen. You can either hold the mouse button, move the mouse pointer over the menu entry that you want to activate, and then release the mouse button, or you can release the mouse button right after pressing it, and then select the menu entry with the left or the right mouse button.

If you press <**Ctrl**>**RM** in the speech signal window or the label window, a special pop-up menu providing edit functions is displayed.

In Figure 3.4 you can see the window menu and the edit menu for the speech signal window.

Brackets: *xassp* provides two brackets which delimit a region. Brackets are displayed as vertical bars in the data area. The region that the brackets define can now be used to play part of the speech signal, to analyse the region separately from the rest of the signal, or to perform cut and paste operations in the speech signal window. Normally, brackets are set onto the next zero crossing to the left of the cursor, but this behaviour can be controlled by the user.

Brackets can be set with **F5** (left bracket) and **F6** (right bracket). To override the default setting brackets on zero-crossings you can use <**Ctrl**>**F5** and <**Ctrl**>**F6** to set the brackets on the exact cursor position. The brackets can be cleared with **Return** or **Del**. Clearing the brackets with **Return** does not work on all machines.

Temporary marker: The temporary marker can be thought of as a way of marking a certain point in time, where a label is to be placed. It is set with <**Shift**>**LM**.

The temporary marker is always set on the exact cursor position. It can be moved onto the next positive or negative zero crossing to the left by pressing **F3** and to the next positive zero crossing to the right by typing **F4**.

Not all of these elements make sense for all window types, e.g., brackets and a temporary marker are not available in the section window, because it has no horizontal time scale.

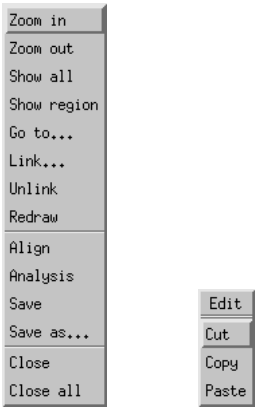

Figure 3.4: *xassp* Window Menu (left) and Edit Menu (right) for the Speech Signal Window

3.4.2 Keyboard and Mouse

xassp windows are controlled by a combination of mouse and keyboard.

When using the keyboard it is important that the window you want to control has the input focus. This is normally achieved by clicking into the title area of the window, or, in some configurations, by simply placing the mouse pointer inside the window. The window that has the input focus usually has a differently coloured border and title area than the other windows.

There are some keys that are used in almost all windows for the same purpose. These are listed in Table A.1. Since the section window has no horizontal time scale only the binding for the `Esc` key applies here.

The mouse buttons are mainly used to play parts of the speech signal. Refer to Table A.2 for a list of commonly used mouse bindings. Again, the commands for playing the speech signal are not available in the section window.

Appendix A provides a complete reference to the key and mouse bindings used in *xassp*.

3.4.3 Pop-up Menus

As explained in Section 3.4.1 each *xassp* window has a pop-up menu, which is called up by pressing <**Alt**>**RM** (see, e.g., the pop-up menu of the speech signal window in Figure 3.4). The pop-up menus of the different windows have a lot of items in common, which will be described first. The special menu items of each window are listed in the sections that deal with the different window types.

Zoom in: By activating this menu item you increase the horizontal (time) resolution that the data are displayed with. This implies that the part of the data that is displayed becomes smaller. This action is called zooming in, because it resembles the zooming performed by a camera.

Figure 3.5: *Go to* Dialog Box and Link Selection Menu

Zoom out: Same as *Zoom in* except that the resolution is decreased and therefore the part of the data that is displayed becomes larger.

Show all: The resolution is adjusted so that all data are displayed at the same time.

Show region: The region that is defined by the two brackets becomes the part of the data that is displayed.

Go to: Jumps to a certain point in time or to a label (see Figure 3.5).

Link: Links different features of the windows displayed on the screen (see Section 3.4.4).

Unlink: Unlinks the current window from the other windows displayed on the screen.

Redraw: Redraws the window contents in case parts of the data were accidentally not drawn.

Print: Displays the *Print* dialog. Please refer to Section 3.7 for more information on printing.

Close: Closes the current window. If you made changes to the data, you are prompted whether you would like to save them.

Close all: Closes all windows.

Since zooming depends on the horizontal time scale, these actions are not available in the section window.

3.4.4 Window Linking

It was mentioned in Section 3.3.6 that it is possible to link windows. This is useful for a better overview of the different data files associated with a particular signal file, for instance when segmenting the speech signal. There are several ways of linking windows. The first is shown in Section 3.3.6: in the file selection box different data types, e.g. speech signal and fzero, are selected. Then the button *Align and link* is pressed. The two windows will be displayed on the screen aligned and linked. By default all window features cited below are linked.

Another way of getting aligned and linked windows is to use one of the configuration buttons directly above the *Selection* field (see Figure 2.1). For example, *Segmental*, activates a speech wave, a sonagram and a label file in separate windows on the screen: these windows are linked and aligned automatically.

Finally, the menu item *Link* (see Figure 3.5) may be used in one of the windows. It has to be chosen if the windows have been called up separately without any configuration. You can also use this menu to unlink one or several window features. If you want all features to be unlinked, select the window menu item *Unlink*.

In the following list all possible links are explained.

Cursor: For several tasks it is necessary that the cursor is at the same point in time in each window. Especially for segmenting and labelling a speech signal it is very important that you can see where you are to correctly position a label.

Zoom: The temporal resolution is the same in all linked windows.

Brackets: The brackets are set at the same point in time in linked windows.

Play: If a non-speech signal window is linked to a speech signal with a play link, you can use the key and mouse buttons listed in Table A.2 in the non-speech signal window to play the signal in the speech signal window.

Scrolling: The starting time for the data display is the same in all windows.

Temporary marker: The temporary marker is set at the same point in time in all linked windows.

3.4.5 Data Logging

xassp provides the possibility to write F_0-values, energy values, formants and labels that are associated with a certain point in time to a log file. To enable data logging you have to activate the *Data logging* button in the *xassp* main dialog. If logging is on, a status window appears in the lower right corner of the screen (see Figure 3.6). It shows which file was selected as the *xassp* log file (the default file is `xassp.log` in the directory *xassp* was started in). Pressing the *Close* button closes the status window and disables data logging. If you select the *File* button, you can change the *xassp* log

Figure 3.6: The *Data Logger* Status Window

file by picking a new file in the file selection box that is displayed. Once data logging is enabled, you can position the cursor at the point in time of interest and press `w` to write the data to the log file. The cursor will now briefly change its shape to a pencil to confirm that data have been logged. If this does not happen, check which window has the focus because this determines the search for data to log. Also, some windows, such as the label and formant lists, do not receive the keyboard command to log data.

Choosing Data for Logging

The data items that are to be logged can be chosen by opening the corresponding windows. E.g., if you want F_0-values to be written to the log file, you have to load an F_0-file with the same base name as the speech signal file, or do an F_0-analysis of the speech signal file. *xassp* always logs data items from those windows that belong to the same speech signal, i.e. if a speech signal window has the focus, and you press `w`, the values from all associated windows are written to the log file. If a non-speech signal window has the focus and `w` is pressed, *xassp* checks whether it contains analysed data. If so, it searches the speech signal window that this window refers to and logs the values from all windows associated with this speech signal. Otherwise, it uses the basename of the file to determine which other windows contain associated data.

Log File Format

xassp log files consist of lines that contain TAB-separated items. The first line of a log file contains the items `XASSP` and `DATALOG`. Each time you press `w` a line is appended to the log file. The items in each line are paired, the first item of a pair is a keyword, the second is the associated value. The keywords that are used in *xassp* log files are listed in Table 3.3.

An *xassp* log file could, e.g., look like this:

```
XASSP   DATALOG
TIME    0.82625 LABEL   ##J     F0      0       FILE    g071a000.r16
TIME    0.9475  LABEL   $'A:    F0      131     FILE    g071a000.r16
TIME    1.09375 LABEL   $T-N    F0      136     FILE    g071a000.r16
TIME    1.31625 LABEL   ##D     F0      0       FILE    g071a000.r16
TIME    1.4575  LABEL   $N+     F0      127     FILE    g071a000.r16
```

Keyword	Description
`TIME`	The point in time the logged data items are associated with
`LABEL`	The label that is associated with the segment containing the chosen point in time
`F0`	An F_0-value in Hz
`RMS`	An energy value in dB
`F`n[a]	The frequency of the nth formant in Hz
`B`n	The bandwidth of the nth formant in Hz
`A`n	The amplitude of the nth formant in dB
`FILE`	The file that is associated with the window in that you pressed `w`

[a] n is to be substituted by an integral number

Table 3.3: Keywords used in *xassp* log files

Adding a Comment to logged Data

If you wish to add some comment to the logged data, press <**Ctrl**>`w` rather than `w`. After the data have been collected, a text window will appear in which you can enter your comment. After pressing **Return** or the *OK* button, the line you entered will be written to the log file after the line with the data. As identifier, the line is preceded by the word `COMMENT` and a TAB.

3.5 Specific *xassp* Window Properties

In this section the different data and the resulting different window types will be explained. These are:

- Speech signal
- Sonagram
- Section
- Labels
- Fzero
- Energy

3.5.1 The Speech Signal Window

In the speech signal window the speech wave of a selected signal file is displayed.

Window Menu

In addition to the general *xassp* menu items listed in Section 3.4.3, you have some special items in the pop-up menu (see Figure 3.4):

Analysis: Different analyses of the speech signal are possible: sonagram, section, energy, fzero, and labels. The *Analysis* dialog box that is opened upon selecting this entry in the pop-up menu is shown in Figure 3.7. You can choose one or more data types. After pressing the *OK* button, for each selected data type an analysis of the speech signal is performed, and the result is displayed in a separate window. If the *Align and Link* button is selected, all *xassp* windows are aligned on the screen (see below for further explanation) and all new analysis windows are linked to the speech signal window.

Furthermore, you can limit the analysis to the region that is defined by the two brackets by selecting the *Region* button. This is not possible if no brackets or only one bracket is set.

There is no analysis routine that generates a label file from a speech signal file, so *xassp* opens an empty label file if *Labels* was selected in the *Analysis* dialog box.

Align: *xassp* places the first window in the upper left corner of the screen, the second directly below the first, etc. Then the windows are arranged in such a way that no *xassp* window is obscured by any other window. The order in which *xassp* stacks the windows is the same as the order in which the windows were opened. This order cannot be changed.

xassp windows are automatically aligned if in opening files a configuration is used (see Section 3.3.6), or if the *Align and Link* button in the *xassp* main dialog is selected when opening a file (see Section 2.2), or if the *Align and Link* button in the *Analysis* dialog box is selected when doing an analysis (see above).

Save: Saves the speech signal under the file name that is displayed in the title area of the window. If this file exists, it is overwritten without warning.

Save as: Saves the speech signal under a different file name. When selecting this menu item you can select a file name and a file format. The file formats you can choose from are: KTH, RIFF-WAV, CSL, and RAW. For further information on speech signal file formats see 3.6.1. If the the file you select exists, you are prompted whether it should be overwritten.

Edit Menu

Apart from playing and analysing the speech signal, it can also be edited. This is important in, e.g., the generation of speech stimuli. For this task you need at least

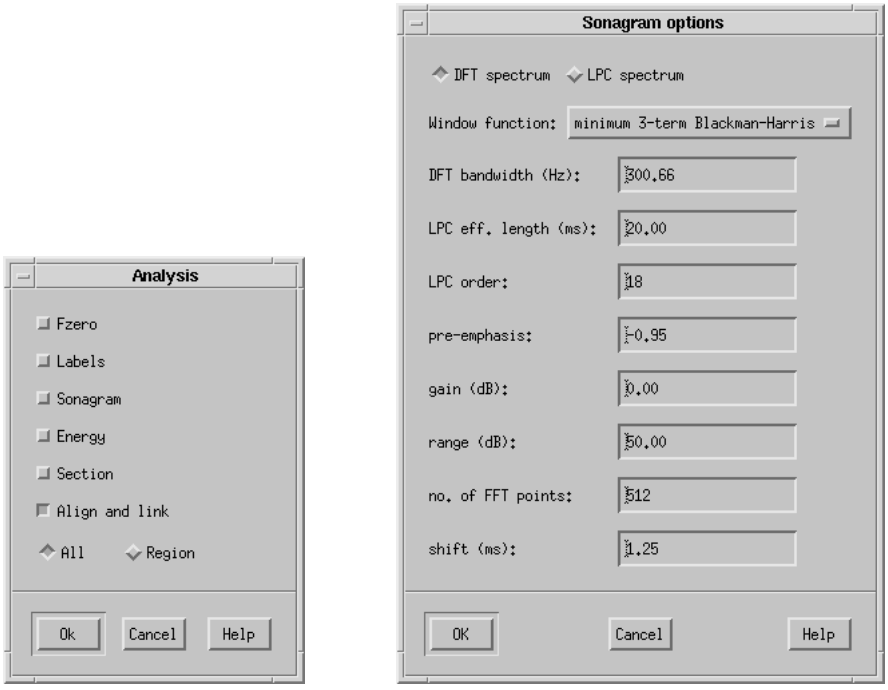

Figure 3.7: Left: Analysis dialog box. Right: Sonagram options

userlevel 30 (see Section 3.1). The edit menu is called up by pressing the combination <**Ctrl**>**RM** (see Figure 3.4). The menu items are set out in the following list:

Cut: Cuts the region that is defined by the two brackets out of the speech signal and saves it in the cut buffer.

Copy: Copies the region into the cut buffer.

Paste: Inserts contents of the cut buffer at the position of the temporary marker. If no temporary marker is present, a new window is opened with the cut buffer contents.

Keys

With the keys **CursorUp** and **CursorDown** it is in addition possible to increase or decrease the amplitude resolution of the speech signal. A list of keys that are available in the speech signal window is given in Table A.3.

3.5.2 The Sonagram Window

One type of spectral analysis of the speech wave results in a sonagram. A sonagram is a time-frequency-amplitude representation of the speech signal.

Window Menu

The window menu of the sonagram window has only one additional item:

Re-analyse: Selecting this menu entry opens the *Sonagram Analysis Options* dialog box (see Figure 3.7). See below for a description of the analysis parameters that can be changed in this dialog box.

Analysis Options

In the *Sonagram Analysis Options* dialog box, which is opened by selecting the menu item *Re-analyse* in the pop-up menu, you can modify the following parameters:

DFT/LPC spectrum: The DFT spectrum with a bandwidth of about 300 Hz corresponds to a wide-band sonagram, one with a bandwidth of about 50 Hz to a narrow-band sonagram. The LPC spectrum shows the spectral envelope of the speech signal, enhancing the formant structure.

Window function: Specifies the window function used during analysis.

DFT bandwidth (Hz): Specifies the bandwidth of the DFT spectrum.

LPC eff. length (ms): Specifies the effective length of the LPC analysis window.

LPC order: Specifies the order of the LPC analysis.

Pre-emphasis: Specifies the pre-emphasis applied to the speech signal before spectral analysis.

Gain (dB): Specifies the gain applied in mapping the spectral levels to grey levels.

Range (dB): Specifies the difference between the highest and lowest spectral level to be displayed.

no. of FFT points: Specifies the length of the FFT used in calculating the spectra; it also sets the vertical resolution to *sampling frequency/number of points* [Hz per pixel].

Shift (ms): Specifies the frame shift for the analysis; it also sets the horizontal resolution to *shift* [ms per pixel].

Please refer to Section 5.4 and to Chapter 6 for further information on spectral analysis.

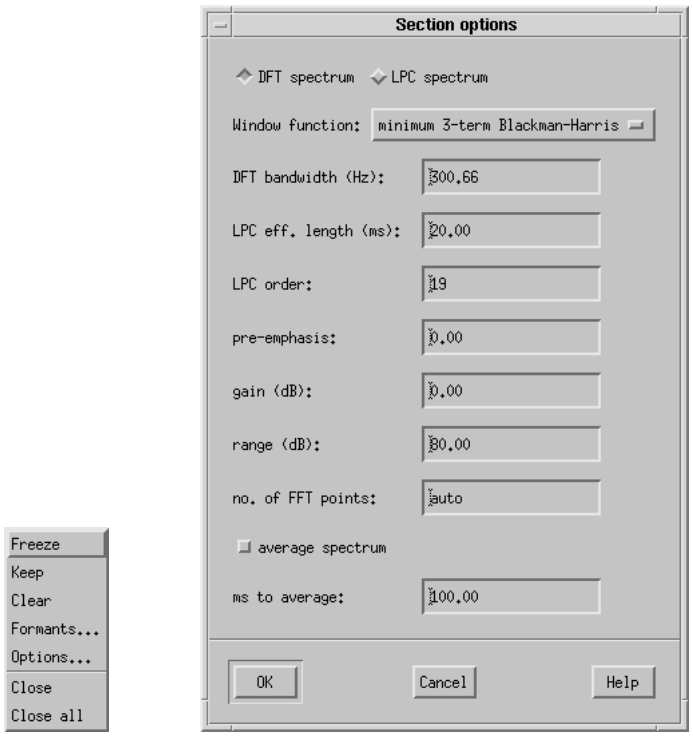

Figure 3.8: The Section Pop-up Menu and Section Options

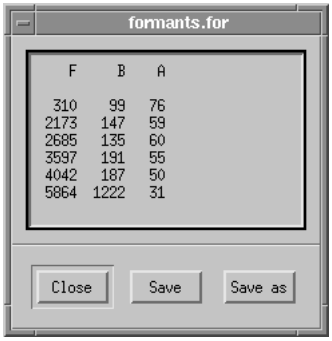

Figure 3.9: The Formants List

3.5.3 The Section Window

The section window shows the spectrum of the speech signal at the current cursor position.

The section window can be opened by selecting *Section* in the *Analysis* menu of the speech signal window, or by the short-cut <**Shift**>s (see also Table A.3 for key bindings in the speech signal window).

Window Menu

The pop-up menu (see Figure 3.8) contains the following entries:

Follow/Freeze: Selecting this menu item toggles the updating of the section by cursor movement in the speech signal window.

Keep: Make a copy of the section that is currently displayed. This copy is then shown in addition to the current section allowing you to compare sections at different points in time or with different analysis parameters.

Clear: Remove the last section that was kept with *Keep*.

Formants: Shows the *Formants Options* dialog box that lets you choose the display of a formant list in a separate window (see Figure 3.9) and the number of formants to display. The formants can only be computed if *LPC* was selected in the options menu.

Re-analyse: Opens the *Section Options* dialog box shown in Figure 3.8. The analysis parameters you can change here are explained below.

Analysis Options

By selecting the menu item *Re-analyse* the *Section Analysis Options* dialog box shown in Figure 3.8 appears. Here you can make the following changes:

DFT/LPC spectrum: The DFT spectrum shows the Fourier spectrum of the speech signal, while the LPC spectrum shows the spectral envelope.

Window function: Specifies the window function used during analysis.

DFT bandwidth (Hz): Specifies the bandwidth of the DFT spectrum.

LPC eff. length (ms): Specifies the effective length of the LPC analysis window.

LPC order: Specifies the order of the LPC analysis.

Pre-emphasis: Specifies the pre-emphasis applied to the speech signal before spectral analysis.

Gain (dB): Specifies the gain applied in determining the highest spectral level to display.

Range (dB): Specifies the difference between the highest and lowest spectral level to be displayed.

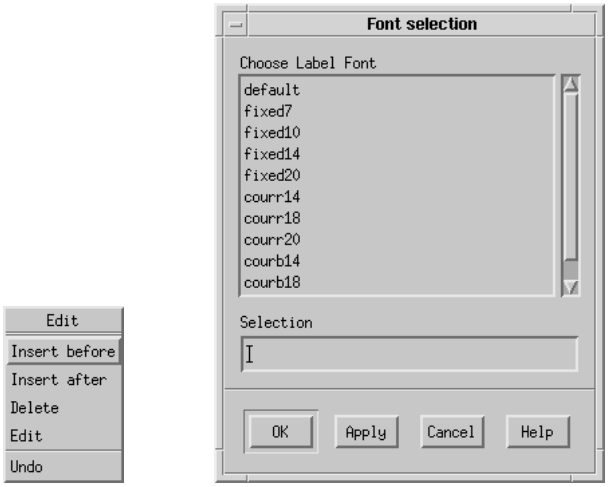

Figure 3.10: Left: Label Edit Menu. Right: *Font Selection* dialog box

no. of FFT points: Specifies the length of the FFT used in calculating the spectra. If set to 0, the program automatically determines the optimal number of FFT points, given the analysis parameters and the width of the display window (the latter to obtain a smooth spectrum). This will also be shown in the option box by the word *auto* in the field for the number of FFT points.

Average spectrum, ms to average: If this button is selected, the average spectrum of a region is calculated. The length of this region is given in the *ms to average* text field. The cursor always defines the centre of the region. Averaging is only possible for the DFT spectrum. It can be simulated for the LPC spectrum by increasing the effective window length.

The section window provides several short-cuts to the items of its window menu; they are listed in Table A.4.

For further information on spectral analysis, see Section 5.4. The subject of formant analysis is covered in Section 5.3. Chapter 6 contains important terms that are used in all analysis descriptions.

3.5.4 The Label Window

The label window lets you assign labels to distinct points in time. This is used to mark the segment boundaries of an utterance or to describe its prosodic structure as in segmental and prosodic labelling, respectively.

Window Menu

The window menu of the label window has the following additional entries:

Show as text: Opens a dialog box containing a label list. In each line of the list the time in milliseconds and the associated label are displayed.

Label font: This menu item opens the *Font Selection* dialog box (see Figure 3.10) that lists the fonts that are defined in the font configuration file (see Section 4.2.2). You can either choose a label font from the list, or you can type a valid *X* font name into the *Selection* field.

Save: Saves the labels to the file that is given in the title of the window. If this file exists, it is overwritten without warning.

Save as: You can save the file under a different file name. If the file you selected exists you are prompted whether to overwrite it.

Edit Menu

With <**Ctrl**>**RM** you get the label edit menu (see Figure 3.10), which has the following entries:

Insert before: Insert a new label before the highlighted label.

Insert after: Insert a new label after the highlighted label.

Delete: Delete the highlighted label.

Edit: Edit the highlighted label.

Inserting a New Label

If a new label is to be inserted at a certain point in time, the temporary marker must first be set onto this point. Then you press <**Ctrl**>**MM** to insert the label at the temporary marker.

If there already are labels at the point in time at which you want to insert a new one, you use the edit menu items *Insert before* and *Insert after*.

Moving a Label

For moving a label to a different point in time, the temporary marker must first be set onto this point. Then place the cursor between the temporary marker and the label to be moved. If the label to be moved has not been set yet, you have to place the cursor after the last label of the file. To actually move the label press <**Shift**>**MM**. To edit the label before moving press <**Shift**>**RM**. You can automatically move the label and mark it as deleted (append a - (minus)) with <**Alt**>**LM**, or mark it as deleted and edit it before moving with <**Alt**>**MM**. An uncertainty marker can be added before moving the label with <**Ctrl**>**LM**.

Keys

The key `h` can be used to insert an aspiration label at the temporary marker, if it is placed directly after a plosive, or a creak label, if it is placed after a glottal stop. If the temporary marker is placed elsewhere, nothing is inserted when typing `h`.

Typing `z` inserts a lengthening label (`$-z:`) at the temporary marker.

Pressing <**Ctrl**>`s` makes the *Go to* dialog box appear on the screen.

<**Ctrl**>`f` jumps to the next occurrence of the label that was last searched for.

<**Ctrl**>`b` jumps to the previous occurrence of the label that was last searched for.

<**Ctrl**>`l` jumps to the last label.

Refer to Table A.5 for a complete list of key and mouse bindings in the label window.

Label Syntax Checks

If *xassp* is run with a user level below 30, a syntax check is performed for each label that is inserted, edited or moved. If the label is not valid, an error message appears and the edit process is aborted.

The following additional checks are done when a label is . . .

deleted: Only prosodic labels, inserted labels or labels with a `kp` prefix may be deleted.

inserted: The inserted label must be a prosodic label, a label that is marked as inserted or a label with a `kp` prefix. Prosodic labels may not be inserted at the temporary marker.

edited: Labels that are marked as inserted may not be edited. To change such a label you must delete and re-insert it.

You may not change the label class (prosodic or segmental), the canonical label or `$` (word-internal) prefixes. You are not allowed to modify the lexical stress markers or the `-` (minus) that indicates that the label has been inserted.

3.5.5 The Fundamental Frequency Window

In this window the fundamental frequency of a given speech signal file is displayed. The window can be called up with the button *Fzero* in the *xassp* main dialog, or with the window configuration *Prosodic*, which gives you speech signal, fundamental frequency and labels, or it can be analysed directly from a speech wave in the *Analysis* dialog box in the speech signal window (select *Analysis* in the pop-up menu of the speech signal window). In each case a separate window appears for the fundamental frequency.

Window Menu

There are several special menu items for this window:

Draw mode: You can choose between three draw modes of the F_0-values: circles, squares and lines.

Logscale: By selecting this menu item you can toggle between a logarithmic and a linear scale.

Set range: In the *Set Range* dialog box that opens when selecting this entry you can specify the minimum and maximum F_0-value that should be displayed. If the *Compute* button is pressed, the minimum and maximum values are taken from the data. Note that this only affects the display, not the analysis.

Re-analyse: Opens the *Fzero Analysis Options* dialog box (see below). This item is only present when the F_0 data were analysed on-line rather than loaded from a file.

Save: Saves the F_0 data in the file that is shown in the title of the window.

Save as: Opens a file selection box to save the F_0 data in a different file. If the file you selected exists you are prompted whether to overwrite it.

Analysis Options

In the *Fzero Analysis Options* dialog box, which you get by selecting the *Re-analyse* menu entry in the pop-up menu of the fundamental frequency window, you can specify the following options:

All/Region If both brackets have been set, you can choose whether to re-analyse the whole signal or only the part in the marked region. Re-analysing a region is particularly useful if you want to correct an octave error (see Section 5.2).

Frame shift (ms): Specifies the frame shift for the analysis.

Maximum F0 (Hz): Specifies the highest F_0-value to be analysed.

Minimum F0 (Hz): Specifies the lowest F_0-value to be analysed.

Noise amplitude: Specifies the minimum speech signal amplitude considered relevant for analysis.

Please refer to Section 5.2 for more information on F_0-analysis.

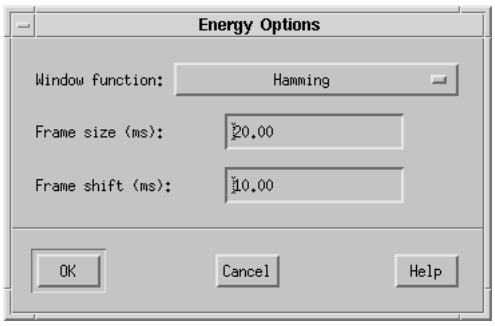

Figure 3.11: Menu item *Re-analyse* in the energy window

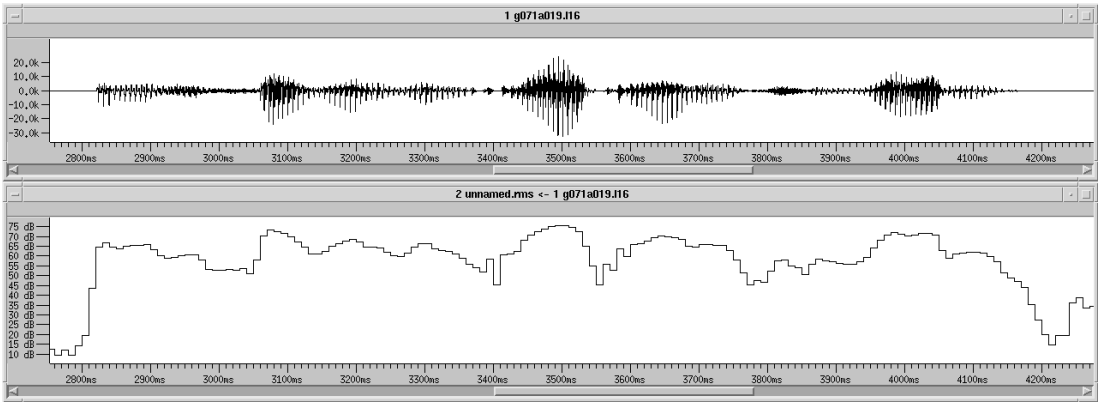

Figure 3.12: Upper window: speech wave, lower window: analysed energy course

3.5.6 The Energy Window

The energy course of a speech wave can be analysed by clicking on *Energy* in the *Analysis* dialog box called up via the pop-up menu of the speech signal window (see Section 3.5.1). The menu item *Align and Link* has to be chosen to display the energy window linked with the speech signal window on the screen (see Figure 3.12).

Window Menu

The following special menu items exist in this window (see Figure 3.11):

Draw mode: You can choose between three draw modes of the energy values: circles, squares and lines.

Re-analyse: Opens the *Energy Analysis Options* dialog box (see below). This item is only present when the energy data were analysed on-line rather than loaded from a file.

Save: Saves the energy data in the file that is shown in the title of the window.

Suffix	File type
.r16, .l16, .shh, .wav, .raw, .syn	speech signal
.f0	fzero file
.mix	label file
.rms	energy file

Table 3.4: File suffixes

Save as: Allows you to select a file in which to save the energy data. If the file you selected exists you are prompted whether to overwrite it.

Analysis Options

In the *Energy Analysis Options* dialog box you can change the following options:

All/Region If both brackets have been set, you can choose whether to re-analyse the whole signal or only the part in the marked region.

Window function: Specifies the window function to be used during analysis.

Frame size (ms): Specifies the frame size for analysis.

Frame shift (ms): Specifies the analysis frame shift.

Please refer to Section 5.1 for a description of the energy analysis and to Chapter 6 for an explanation of the terms that are used in the analysis descriptions.

3.6 File Handling

xassp can load data to be displayed from and save data to files. The data types that can be loaded and saved are

- Speech Signal
- Fundamental Frequency
- Energy
- Labels

In the following sections the different file formats that are used in *xassp* are explained.

3.6.1 Speech Signal Files

xassp can handle speech signal files that have one of the formats that are explained in this section.

KTH File Format

Files in KTH format consist of an ASCII header followed by binary data. The reason to use an ASCII header is that it can be viewed with a `more` or `less` command or a simple text editor such as `vi`.

The header of a KTH file normally looks like this:

```
head=1024<CR><LF>
file=samp<CR><LF>
data=int16<CR><LF>
msb=first<CR><LF>
nchans=1<CR><LF>
sftot=16000<CR><LF>
length=203722<CR><LF>
=<CR><LF><SUB><EOT>
<NUL>
<NUL>
⋮
```

The `<CR>` is a carriage return character, `<LF>` is a line feed character, and `<NUL>`, `<SUB>` and `<EOT>` are special ASCII characters. The dots indicate that the `<NUL>` characters are repeated until the end of the header is reached.

The header consists of simple keyword-value-pairs. The most interesting ones are `head`, which gives the header length in bytes, `sftot`, which gives the sampling frequency in samples per second, and `msb`, which gives the byte order.

CSL File Format

The CSL format is used by the *Computerised Speech Lab*, a combination of hard- and software for sampling and analysing speech data. It is a pure binary format.

WAV File Format

The WAV file format is widely used and has therefore been included in *xassp*. It can be read by almost any application that plays or manipulates sounds.

3.6.2 Fundamental Frequency

xassp knows only one file format for the fundamental frequency. It is an ASCII format. Each line consists of the time and the associated F_0- value separated by whitespaces (spaces, TABs, etc.). A part of a typical F_0 file looks like this:

```
⋮

0.840000        214.395721
0.850000        223.070328
0.860000        226.532837
0.870000        229.000473
0.880000        221.581589
0.890000        209.492630
0.900000        190.813736
0.910000        168.316666
0.920000        151.206406
0.930000        139.145554
0.940000        130.742325
0.950000        124.183624
0.960000        118.041351
0.970000        116.924339
0.980000        115.094978
⋮
```

xassp fundamental frequency files can optionally contain a header line, which must be the very first line of the file. It contains a file type identifier (`XASSP FZERO`) and the sampling frequency.

3.6.3 Energy

The file format for energy data is exactly the same as for fundamental frequency, except for the header, in which the file type identifier `XASSP ENERGY` is used instead of `XASSP FZERO`.

3.6.4 Labels

xassp recognises two label file formats, MIX used internally at *ipds* and SAM, the label file format found on CD-ROMs#1–4 (IPDS 1994, 1995, 1996, 1997). A MIX file starts with a header, which contains the orthographic and the canonical form. The lines following the header consist of the identifier `FR`, the sample number, the label, the time in centiseconds and the time in seconds.

```
TEXT:
TIS010.
das w}rde mir ganz gut passen . also <P> Dienstag ,
drei~igster , Mittwoch , erster ? <{hm> ja , w}rd'
ich sagen . das k|nnen wir ja dann so machen , dann
werd' ich da meine anderen Termine absa<;T>
PHONET:
  D A S+  V Y r D E0+  M I: r+  G 'A N T S  G 'U: T
  P 'A S E0 N  .  Q A L Z O:+  p:  D 'I: N S T A: K
  ,  D R 'EI S I CH S T r  ,  M 'I T V O X  ,
  Q 'E: r S T r  ?  v: Q [: M  J 'A:  ,  V Y r D E0+
  Q I CH+  Z 'A: G E0 N  .  D A S+  K  N E0 N+
  V I: r+  J A:+  D A N+  Z O:+  M 'A X E0 N  ,
  D A N+  V E: r D E0+  Q I CH+  D A:+  M EI N E0+
  Q 'A N D r R E0 N  T E r M 'I: N E0  Q 'A P #Z "A: ;

CT 1
FR      1010 #c:              7     0.06306 sec
FR      1010 #-s:             7     0.06306 sec
FR      1198 #-h:             8     0.07481 sec
FR      3549 #&0             23     0.22175 sec
FR      3549 ##D             23     0.22175 sec
FR      5091 $-d             32     0.31812 sec
FR      5488 $A              35     0.34294 sec
FR      6130 $S+             39     0.38306 sec
⋮
```

The time in centiseconds and the time in seconds are optional. *xassp* accepts, e.g., the following file:

```
TEXT:
GEP002.
<#> dann treffen wir uns um neun Uhr <#Rascheln> <Ger{usch>
<A> und <Z> machen <Z> <{h> bis maximal zw|lf <Ger{usch> .
PHONET:
  :k  D A N+  T R 'E F E0 N  V I: r+  Q U N S+
  Q U M+  N 'EU N  Q 'U: r  :k p:  g:  h:  Q U N T+ z:
  M 'A X E0 N z:  v: Q [:  B I S+  M A K S I M 'A: L
  T S V ' L F  g:  .

CT 1
FR 1600000 #:k
```

```
FR 1600000 #c:
FR 1600000 ##D
FR 1600000 $A
FR 1600000 $N+
FR 1600000 ##T
FR 1600000 $R
FR 1600000 $'E
FR 1600000 $F
FR 1600000 $E0
FR 1600000 $N
FR 1600000 ##V
FR 1600000 $I:r+
FR 1600000 ##Q
FR 1600000 $U
```

⋮

This file is a so-called prototype file. It contains labels that have not yet been associated with specific points in time. These labels are then placed at the segment boundaries of the corresponding speech signal in the process of segmental labelling.

The SAM label file has a marginally different structure from that of MIX and the labels are in modified SAMPA (Wells et al. 1989). As with the MIX format, the SAM format contains an orthographic representation of the signal, a canonical transcription and individual lines containing time stamps and labels. In addition, following the canonical transcription, there is a variant transcription which represents a concatenation of the labels. Lines containing the words oend, kend and hend serve to delimit the various chunks. The following is taken from the beginning of a SAM format label file:

```
g083a007.s1h
KAK007: <A> das k"onnen wir dann gleich dort machen .
        ja , das ist ja praktisch . <P> und vielleicht
        den Freitag vorher gleich<Z> <P> die Besprechung
        daf"ur machen .
oend
  h:  d a s+  k 9 n @ n+  v i: 6+  d a n+  g l 'aI C
  d 'O 6 t  m 'a x @ n  .  j 'a:  ,  d a s+  Q I s t+
  j a:+  p r 'a k t I S  .  p:  Q U n t+  f I l 'aI C t
  d e: n+  f r 'aI t a: k  f 'o: 6 #h "e: 6  g l 'aI C z:
  p:  d i:+  b @ S p r 'E C U N  d a f 'y: 6  m 'a x @ n  .
kend
  c:  h:  %d -h a s+  k -h 9 n-m @- n-+  v i:6-6+
  d a n-N+  g l 'aI C  d -h 'O6 t  m 'a x @ n  .  c:
  j 'a:  ,  d -h a s+  Q- I s t-+  j a:+
  p r 'a k -h t -h I S  .  c:  p:-h:  Q- -q U- n t-+
```

```
  f I- l 'aI C t -h -z:  d -h e: n+  f r 'aI t -h a: k -h
  f 'o:6 #h "e:6  g -h l 'aI C z:  p:  d -h i:+
  b @ S p r 'E C U N  d -h a f 'y:6  m 'a x @- n-N  .
hend
        1 #c:
        1 #h:
    11985 ##%d
    12640 $-h
    12823 $a
    13412 $s+
    15054 ##k
    15715 $-h
```

⋮

3.7 Printing

xassp provides the possibility to generate PostScript output from the data displayed on the screen. You can either configure the parameters for each window (position, size, font, etc.) yourself, or you can use configurations that define parameters and specify the layout for printing.

Choosing the window menu entry *Print*, which is present in all window menus, opens the *Print* dialog. Here you can choose, whether to send the PostScript output directly to the printer or to save it in a file. Further you can specify a print command and an output file in separate text fields.

Clicking the *Select Windows* button opens the *Select Windows* dialog box, where you can select windows, change their parameters and choose from different configurations, which are defined in the *xassp* print configuration file.

Clicking the *Page Options* button opens the *Page Options* dialog box, where you can change the page size, margins and the page orientation (landscape or portrait).

Press the *Print* button to generate the PostScript output. The *Cancel* button closes the dialog box.

3.7.1 Modifying Page Options

In the *Page Options* dialog box you can modify the page orientation, page size and change the margins. Additionally, you can choose whether to generate EPS (Encapsulated PostScript) output, which can then, e.g., easily be included in TEX documents.

You can choose between the page orientations *Landscape* and *Portrait* by selecting the corresponding toggle button in the *Page Orientation* frame.

The page size can be changed either by selecting one of the pre-defined size from the *Paper Size* list or by selecting *Custom* in the *Paper Size* list and specifying the paper size in the *Width* and *Height* text fields (the values must be given in centimetres).

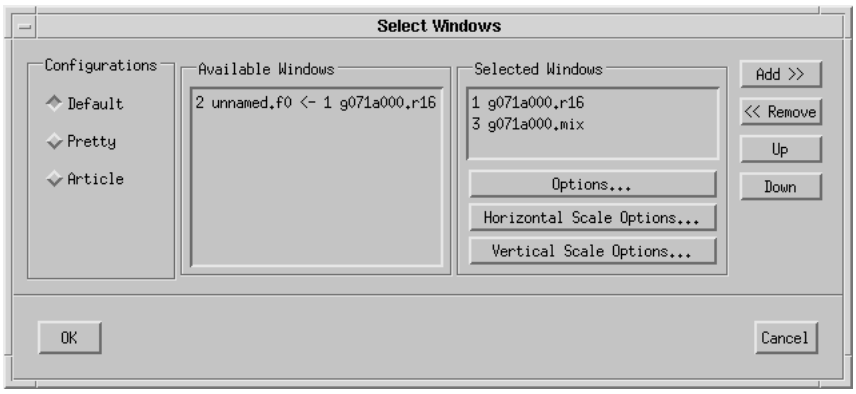

Figure 3.13: *Select Windows* dialog box

In the *Margins* frame you can specify the values for the left, right, top and bottom margins. The values must be given in centimetres. All position values in the option dialog boxes are relative to the margins.

3.7.2 Selecting Windows

If you press the *Print* button in the *Print* dialog box without having selected any windows for printing, *xassp* automatically selects all open windows. To choose the windows that are to be printed yourself you have to open the *Select Windows* dialog box and add the desired windows to the *Selected Windows* list. To do this you select one or more windows from the *Available Windows* list and click the *Add* button, which is located near the right edge of the dialog box. The selected windows are then moved from the *Available Windows* list to the *Selected Windows* list. You can remove windows from the list by selecting them in the *Selected Windows* list and clicking the *Remove* button.

You can change the order of the selected windows by selecting one window in the *Selected Windows* list and clicking the *Up* or the *Down* button to move the selected window up or down in the list. The window order in the *Selected Windows* list is important if you let *xassp* compute the window positions, i.e. if you have selected a configuration.

3.7.3 Choosing a Print Configuration

Print configurations let you define parameters and layouts for printing so that you do not need to change the parameters yourself each time. You can choose a configuration by selecting a toggle button in the *Configurations* frame which is located in the *Select Windows* dialog box. Choosing a configuration changes the window parameters according to the specifications in the *xassp* print configuration file. You can manually change them afterwards. These changes are persistent as long as you do not select a

different configuration.

If you choose the special configuration *Default*, which is not defined in the *xassp* print configuration file, all parameters are set to their default values.

3.7.4 Modifying Window Parameters

xassp lets you specify almost every parameter that can be defined in a configuration. To change the parameters for a certain window you have to select it in the *Selected Windows* list and press the *Options* button if you want to change the window parameters, the *Hscale Options* button if you want to change the parameters of the horizontal scale for this window, or the *Vscale Options* button if you want to change the parameters of the vertical scale for this window.

You can change the following parameters in the option dialog boxes:

- position (horizontal (X) and vertical (Y)),
- size (width and height),
- title,
- title font,
- whether to print a border,
- border width,
- line width (only Wave, Fzero, Energy and Label windows),
- line color or grey value (only Wave, Fzero, Energy and Label windows),
- label font (only Label window),
- print style (only Fzero and Energy windows).

The following parameters can be changed in the options dialog boxes for horizontal and vertical scales:

- font for tick labels,
- minimum space between two tick marks,
- margin between tick labels and tick marks.

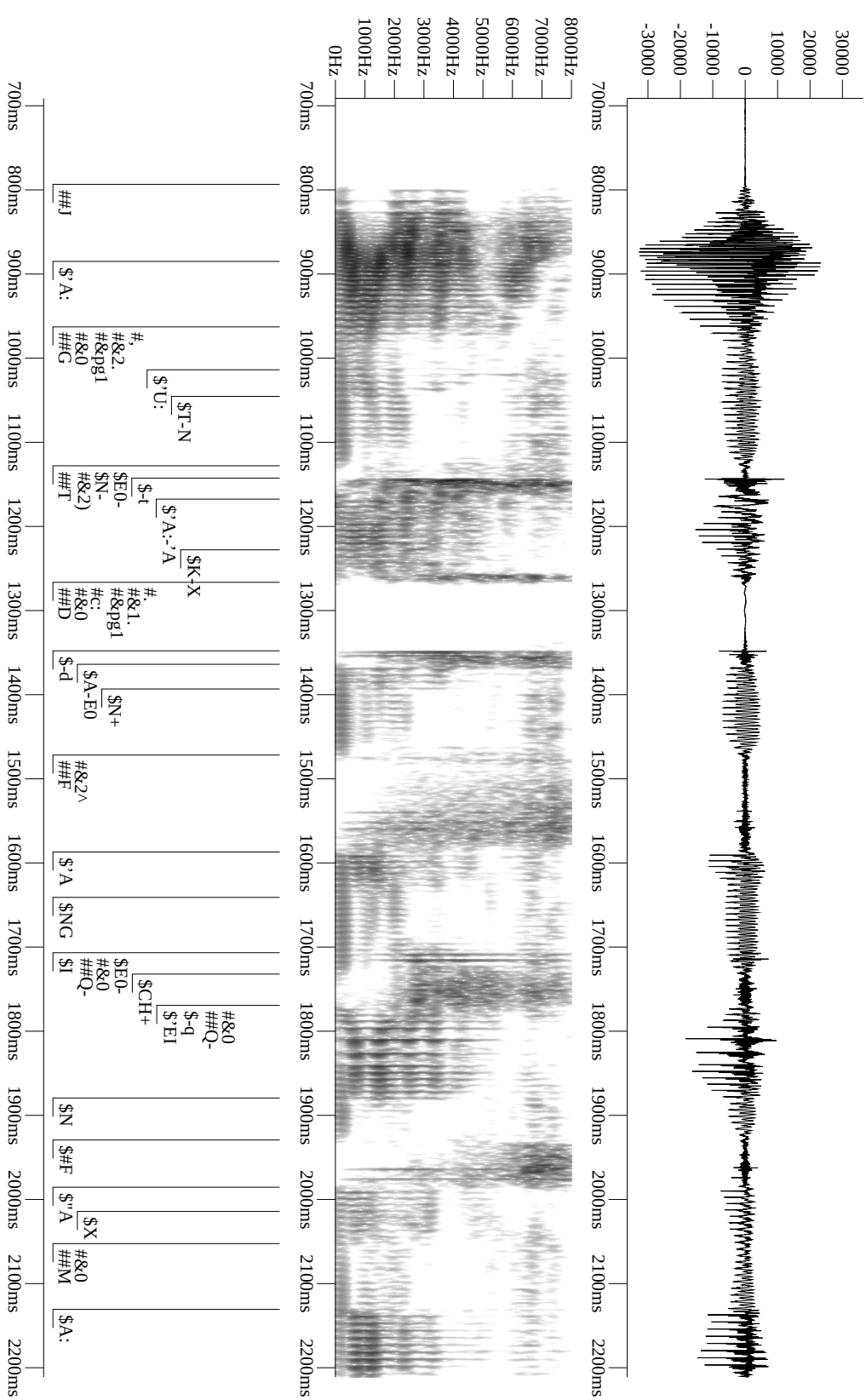

Figure 3.14: Print output generated with *xassp*

Chapter 4

Configuring *xassp*

Since *xassp* is an application that runs under the *X Window System*, you can make use of the numerous possibilities of modifying *X* resources defined by *xassp*. With these you can control the appearance of *xassp* as well as parameters for the analyses *xassp* is capable of. You can also easily change key and mouse bindings.

Because of the complexity of *xassp*, *X* resources alone are not sufficient to control every aspect of *xassp*. Therefore, *xassp* supports three more configuration files to let the user define the buttons that are to appear in the *xassp* main dialog and the windows to be opened if one of the buttons is selected. Furthermore, you can specify default fonts to choose from, and frequent labels that are presented in the *Edit* dialog box in the label window.

4.1 X Resources

To be able to modify *X* resources, it is important that you understand how the *xassp* widgets are structured and which resources they define. In Figure 4.1 you can see what elements (widgets) a typical *xassp* window containing a speech signal consists of. Additionally, an *xassp* window is shown in Figure 4.2 with the designations used in Figure 4.1.

As you can see, every widget in an *xassp* window has a name and a class. If you

Window Type	Prefix
Speech Signal	wave
Fzero	fzero
Labels	labels
Sonagram	sonag
Section	section
Energy	energ

Table 4.1: Widget Name Prefixes

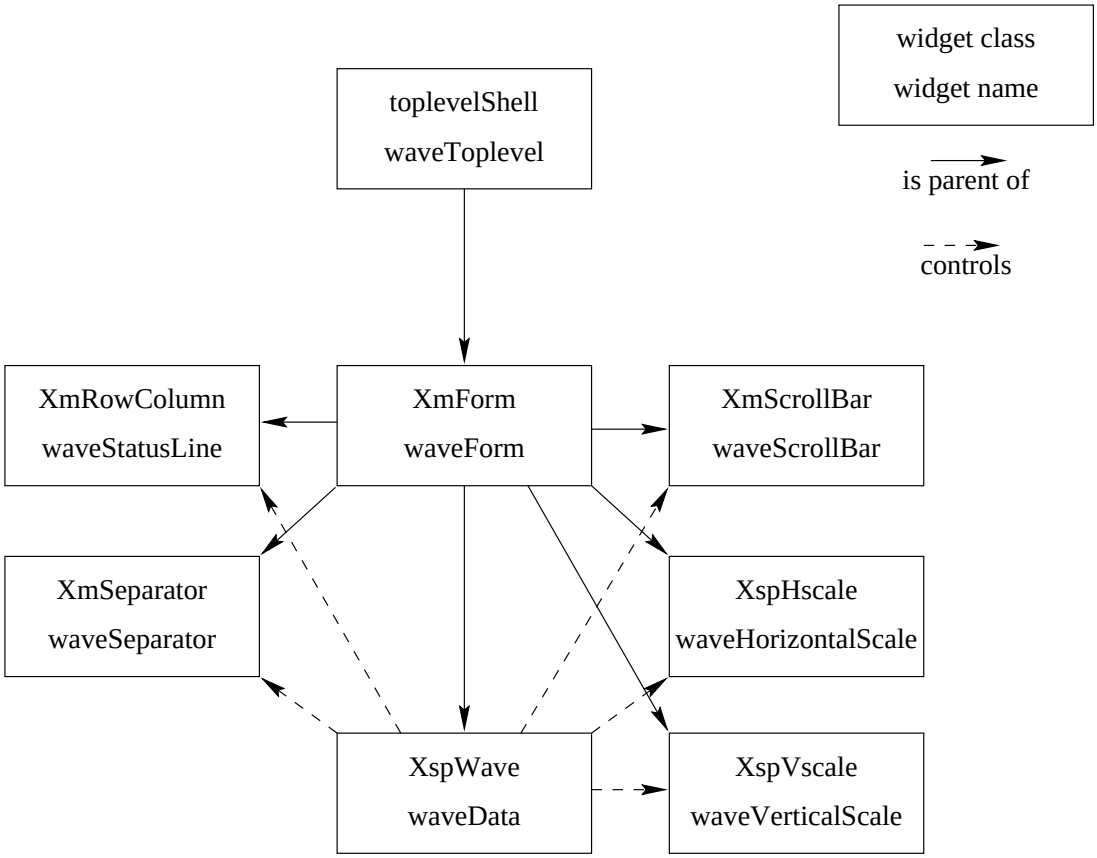

Figure 4.1: *xassp* window structure

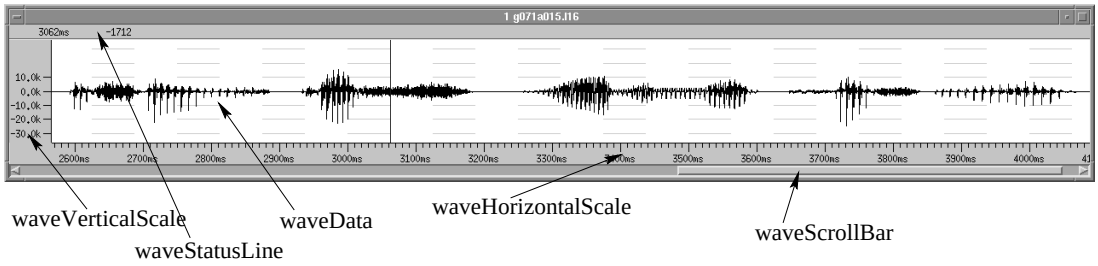

Figure 4.2: *xassp* wave window

refer to the widget class, all widgets belonging to this class are affected when changing some resource. By specifying the widget name, you are able to change resources for all widgets that are elements of the specified window type. The widget names consist of a prefix like *wave* and the actual name. Table 4.1 tells you which prefix to use for each window type.

X resources can be changed in the *.Xdefaults* file in your home directory. If the resource you would like to edit is not present in the file, you can copy it from the *X* application defaults file `/usr/lib/X11/app-defaults/Xassp` where the de-

faults for *xassp* resources are specified. The resources you can change and their possible values are listed in the manual pages for the *xassp* widgets, as well as in appendix B. It is also useful to have a look at the *X* manual page, especially the section called RESOURCES.

Each resource line in your *.Xdefaults* file changing an *xassp* resource normally has the following form:

```
Xassp*<widget name or class>.<resource>: <value>
```

If you want to modify the background colour of all widgets containing speech signals (widget class *XspWave*), you type:

```
Xassp*XspWave.background: white
```

If you want to change the font of the tick labels in the horizontal scale widget of all label windows (widget name labelsHorizontalScale), you need to add the following line:

```
Xassp*labelsHorizontalScale.labelFont: 9x14
```

To learn more about font specifications, you should once more consult the *X* manual page (section FONT NAMES).

To alter the default value for the frame shift used in the calculation of a sonagram, you will have to write:

```
Xassp*XspSonag.shift: 5
```

After modifying your *.Xdefaults* file, you have to run the command

```
xrdb ~/.Xdefaults
```

to make your changes take effect.

4.2 User Configuration Files

The main purpose of the configuration files described in the following sections is to cover configurable aspects of *xassp* that cannot be sensibly defined as resources. The files containing information about fonts and labels are not very complicated, and will possibly be included in the main configuration file in the future.

xassp looks for configuration files in two places. The first is the default global configuration directory (`/usr/local/lib/xassp`), the second is the subdirectory `.xassp` of the user's home directory. (**NOTE**: Using a main configuration file in the global configuration directory has not yet been implemented). The files in the user's home directory override any global defaults.

Option	Description
autogain	If set to *yes*, gain the speech signal to maximum amplitude
hscale	If set to *yes*, display horizontal scale
scrollbar	If set to *yes*, display scroll bar
smallscale	If set to *yes*, display small scale
statusline	If set to *yes*, display status line
vscale	If set to *yes*, display vertical scale

Table 4.2: *xassp* options

4.2.1 The Main Configuration File

The main configuration file is named `config`. Here you can specify the configuration buttons that appear in *xassp* main dialog, and which windows should be opened if one of them is selected. Furthermore, you can change global and configuration-specific defaults.

The file starts with a section in which you can specify values for global options. If you change options at this point, every window that you open in *xassp* is affected. An option consists of the option name and the value you want the option to take. At the moment you can specify *yes* or *no*. Refer to Table 4.2 for a list of valid options.

If you do not want the status line to be displayed, you have to insert the following line at the beginning of the configuration file:

```
statusline no
```

After the global options section you can define an arbitrary number of configurations, i.e. groups of window types that can then be opened by selecting the corresponding configuration buttons. These definitions have the following form:

```
<identifier> {
[ <option> <value> ]
...
[ <window type> (<suffixes>), ]
...
}
```

The names to be used for the window types are identical to the prefixes listed in Table 4.1. The options that you specify affect only the windows that are defined after the option line. If you specify multiple suffixes for a window type, you must separate them with commas. A suffix must always include a period. If you want the data of a certain window to be computed from the data of another, you have to specify an asterisk followed by the number of the window from which data are to be taken. In this case, the window number is the number of the window in this special configuration.

A simple configuration for segmental labelling could look like this:

```
Segmental {
  wave (.l16,.r16),
  sonag (*1),
  labels (.mix)
}
```

As you can see, the sonagram is computed using data from the first window in the configuration, which is the speech signal window.

If you want only one status line and one scroll bar, you can use the following:

```
Segmental {
  scrollbar no
  wave (.l16,.r16),
  statusline no
  sonag (*1),
  scrollbar yes
  smallscale yes
  labels (.mix)
}
```

In this configuration the scroll bar is only displayed in the label window, and the status line is only present in the speech signal window. Additionally, a small scale is created above the scroll bar in the label window.

4.2.2 The Font Configuration File

The font configuration file is named `fonts` and resides in the global configuration directory or the subdirectory `.xassp` of the user's home directory. It contains a list of fonts that the user can choose from in the *Font Selection* dialog box in the label window. It maps user-defined font names to *X* font names. Refer to the *X* manual page for information on *X* fonts.

Each line of the file contains the user-defined name and the *X Windows* font name separated by a tab. It is important, that no additional whitespaces appear in the file.

To map the name *default* to a 24 point Courier font, you just add the following line:

```
default<TAB>-*-courier-medium-r-*-240-*
```

Be aware that a font that you give the name *default* is not automatically taken as a default font in the label window. If you want to change the default for this, you have to modify the corresponding resource (`Xassp*XspLabels.labelFont`).

4.2.3 The Label Configuration File

The labels configuration file is named `labels` and can be found in one of the default directories (global configuration directory or directory `.xassp` in the user's home directory). It contains labels that are offered in the edit dialog box in the label window. Each line of the file can contain an arbitrary number of labels, which must be separated by TAB characters. Additional whitespaces (blanks, line breaks, etc.) are not allowed.

In the edit dialog box in the label window, each line in the configuration file is shown as a row of buttons that the user can select. This has the same effect as typing the label and pressing the *OK* button (see Section 3.5.4).

To make *xassp* create the buttons that are shown in Figure 2.4 your label configuration file must contain the following:

```
#&pgn<TAB>#&pg/<TAB>#&pg;
#&hp<TAB>#&rp<TAB>#&rm
#&0<TAB>#&1<TAB>#&2<TAB>#&3
$&0<TAB>$&1<TAB>$&2<TAB>$&3
#&0.<TAB>#&1.<TAB>#&2.
#&,<TAB>#&?<TAB>$''
#&0.,<TAB>#&1.,<TAB>#&2.,
#&0.?<TAB>#&1.?<TAB>#&2.?
$-ma<TAB>$-~<TAB>$-z:
#-s:<TAB>#-g:<TAB>#-h:
##-:k
```

4.3 Administrative Configuration Files

4.3.1 The *xassp* Users File

The default path to this file is `/etc/xassp_users`. It contains a list of users that are allowed to use *xassp*. For each user it defines the default user level and the maximum user level.

Each line in this file has the following structure

```
<user>:<default user level>:<maximum user level>
```

If, e.g., the user `az` has the following entry

```
az:10:10
```

he or she receives a default user level of 10, which cannot be increased by any command line parameters. The user `az` can perform all tasks that are needed for prosodic labelling.

If this user wants to do segmental labelling as well, the entry will have to look like this:

```
az:10:20
```

This allows the user to increase the default user level to a maximum user level of 20, and so to perform all tasks that are necessary for segmental labelling.

Chapter 5

Analyses

This chapter gives an overview of the analyses included in *xassp*. The descriptions are kept brief for the general analyses, details of which can be found in speech signal processing textbooks like Markel and Gray (1976). The F_0-algorithm is described in more detail because it is a method developed at IPDS. Most signal processing terms used below are described in more detail in Chapter 6. Those of you who are not too familiar with signal processing can find good introductions in Rosen and Howell (1991) and Ladefoged (1996). The latter book also includes chapters on Fourier analysis, digital filters and LPC analysis. Beware, however, that especially these chapters unfortunately contain many typing errors.

5.1 Energy Analysis

For showing the course of the signal strength over time, an energy analysis can be performed. As a measure of energy the Root Mean Square (RMS) value is calculated. In formula:

$$s_{\text{RMS}} = \sqrt{\frac{\sum_{n=0}^{N-1} (w_n s_n)^2}{N}}$$

where s is the sampled signal, w the window function, and N the frame size.

You are advised to use a non-rectangular window function and a window overlap of about 50% for a smooth curve. The window function is not very critical and is by default set to Hamming. Defaults for frame size and frame shift are 20 ms and 10 ms, respectively.

5.2 F_0 or Pitch Analysis

xassp contains a version of the Schäfer-Vincent periodicity detector (Schäfer-Vincent 1982,1983). This detector operates on local extrema (minima and maxima) in the speech signal as potential period markers. Triplets of extrema of the same type are evaluated to mark two adjacent periods (a period twin). If a twin is detected, its period durations are compared to those of previously detected twins. Adjacent or partly overlapping twins with comparable period durations are linked to chains (cf. F_0-tracks). If a chain exceeds a certain length (see point 2. below) and there exists no other, longer, chain with shorter period durations (this to suppress octave errors), the stretch covered by that chain is called voiced. For each sample within the stretch, an F_0 value is calculated on the basis of the period markers stored in the chain. The same is done for the samples within the twins that are subsequently appended to this chain. Samples between voiced stretches are called unvoiced and receive an F_0 value of 0. Note that more than one chain may exist at a time, but that only one of them may declare the stretch covered by it to be voiced. Chains die out when no more twins can be appended to them. At a final stage, the F_0 values are re-sampled at the frame rate. If at least half the samples within a frame have been declared voiced (have a positive F_0 value), the frame is assigned the mean of the positive F_0 values. Otherwise it obtains an F_0 value of 0.

The following properties of the algorithm should be kept in mind:

1. In somewhat *difficult* voiced stretches (irregularity in the initial part of a voiced stretch and voiced fricatives), the algorithm has a tendency to declare the stretch unvoiced.

2. Conditions for a chain to mark a voiced stretch include a minimum duration of the stretch of 30 ms. Also, the chain must contain at least three periods and three twins. Short stretches of voicing (e.g. in the vowel of *ich*) may therefore go unnoticed. If possible, select the analysis interval such that it begins and ends in an unvoiced stretch.

3. Creak may be declared voiced or unvoiced, depending on its regularity and duration.

4. The algorithm is optimised for speech signals and may therefore fail on some types of signals such as slightly noisy square waves.

User-definable parameters are *minFzero*, *maxFzero*, *noiseAmp*, and *frameShift*.

***minFzero*:** the lowest F_0 to be evaluated (default 50 Hz). Decreasing *minFzero* slows down processing but has generally no adverse effect on the results. Increasing *minFzero* speeds up processing but has no effect on the results as long as it is lower than the actually lowest F_0 in the signal.

***maxFzero*:** the highest F_0 to be evaluated (default 600 Hz). Decreasing *maxFzero* speeds up processing. However, since fewer potential periodicity markers will be evaluated, it may lead to voicing errors. Decreasing *maxFzero* may be necessary to suppress an octave error. Contrary to most other F_0 analyses, *maxFzero* should *not* be set to slightly above the *correct* F_0 but rather to slightly below its octave (i.e. the lowest erroneous value). Increasing *maxFzero* slows down processing, may in some cases give somewhat better results, but may also lead to octave errors.

***noiseAmp*:** the amplitude threshold for an extremum to be considered a potential period marker (default 16 at 16 bit signal resolution). Extrema with amplitudes below this threshold are assumed to be due to background noise or weak unvoiced signals. Decreasing *noiseAmp* slows down processing and may in some cases lead to erroneous detection of periodicity in noisy stretches. Increasing *noiseAmp* speeds up processing but has no deteriorating effect on the results as long as it is lower than the actually lowest extremum in voiced stretches.

***frameShift*:** determines the rate at which F_0 values are produced at the last stage, namely once per *frameShift* ms. The default value is 10 ms for proper alignment with other analysis data, such as energy contours. Decreasing *frameShift* may in extreme cases lead to gaps in the F_0 track. These may, for example, be caused by a phase shift in the transition from a vowel to a nasal. Note that *frameShift* may be reduced to one sample, because the F_0 values are determined at the sampling rate of the signal. Increasing *frameShift* mainly results in the voicing boundaries becoming increasingly inaccurate. Very short stretches that were declared unvoiced (e.g. in the occlusion of lenis plosives) may therefore disappear because the frames extending over such stretches now contain more voiced samples, resulting in the frames becoming declared voiced.

5.3 Formant Analysis

If an LPC spectrum is selected in Section, a formant analysis may be invoked. In this analysis, local peaks are searched in the LPC spectrum. By parabolic interpolation through the three adjacent FFT points that define such a peak, the peak frequency, its bandwidth and amplitude are estimated. These values will be listed in a dialog box and may be logged in a file. How many peaks will be searched for, can be defined in the section pop-up menu under *Formants*. There, you will also find a toggle button for switching formant calculation on or off. The formant data will continuously be updated with a change in the spectrum in the section window.

When analysing formants, you are advised *not* to set the number of FFT points to *auto* (or 0) because this would result in the formant data becoming dependent on the width of the Section display window. The number of FFT points should rather be set such that a spectral resolution of between 20 and 50 Hz is obtained (e.g. 512 for a

sampling rate of 16 kHz). In general, but specially if you are interested in formant amplitudes, you should set *Pre-emphasis* to 0 and choose an odd LPC order to have one coefficient free for modelling the overall spectral tilt (see Chapter 6).

5.4 Spectral Analysis

Underlying both Sonagram and Section are two spectral analyses, viz. calculation of the Fourier spectrum of the signal via the Discrete Fourier Transform (DFT) and calculation of the spectral envelope of the signal via an LPC analysis. In both cases, the signal is first pre-emphasised and multiplied by the window function specified. For the DFT spectrum, the size of the window follows from the specified bandwidth with a correction for the window function used. For the LPC spectrum, it follows from the specification of the effective length of the window and the window function (see Chapter 6).

The DFT spectrum is then calculated in a straight-forward way using an FFT. For the LPC spectrum, the LPC coefficients are calculated from the windowed signal. From them, the inverse LPC filter is constructed and an FFT is applied to obtain the spectrum of the transfer function of this filter. In both cases, the FFT coefficients are converted to spectral power levels. The parameters *gain* and *range* determine how the power levels will be mapped on your screen: *range* defining the difference between the lowest and the highest level to be displayed; *gain* determining the highest level relative to an internal reference.

If both a sonagram and a section are displayed for a speech signal, setting the analysis parameters for the section to exactly the same values as those for the sonagram will result in a true cross-section of the sonagram.

Chapter 6

Definition of Signal Processing Terms

Bandwidth: The bandwidth parameter determines which details will be resolved in a spectral analysis. Generally, there is a trade-off between the bandwidth (frequency resolution) and the temporal resolution: The better the frequency resolution (smaller bandwidth) the worse the temporal resolution. For display purposes such as sonagrams and sections, we take the product of the bandwidth and the effective window length to be 1. Thus a bandwidth of 400 Hz corresponds to a temporal resolution of 2.5 ms. In a frequency detection task, such as a formant or an F_0-analysis, it is more appropriate to use a product value of 2. In order to analyse F_0 values down to 50 Hz, for example, the effective window length in a frequency-domain analysis must thus be at least 40 ms.

Typical values for a sonagram are about 300 Hz (wide-band analysis in which the harmonics are not resolved) and 50 Hz (narrow-band analysis showing the harmonics as horizontal bars). Note that for very high-pitched voices, a 300 Hz bandwidth may still be too narrow and need to be increased to remove harmonics. There is a lower limit to the bandwidth, determined by the window function and the number of FFT points. If you need a smaller bandwidth than this limit, choose a less compact window function (e.g. a Hamming window) or increase the number of FFT points.

Discrete Fourier Transform (DFT): As the name implies, the DFT is the Fourier transform for discrete signals such as sampled speech signals. Usually, a short-term DFT is calculated. For N input samples, the DFT delivers N (complex) Fourier coefficients. Thus, the frequency resolution of the DFT equals the sampling frequency divided by N. For example, a 512 point DFT at a sampling rate of 16 kHz has a frequency resolution of about 31 Hz. The DFT is usually calculated using the Fast Fourier Transform (FFT). Note that calculating a short-term DFT violates the requirements for applying a Fourier transform. The signal must therefore be multiplied by a tapering window function to reduce spectral distortion due to this violation.

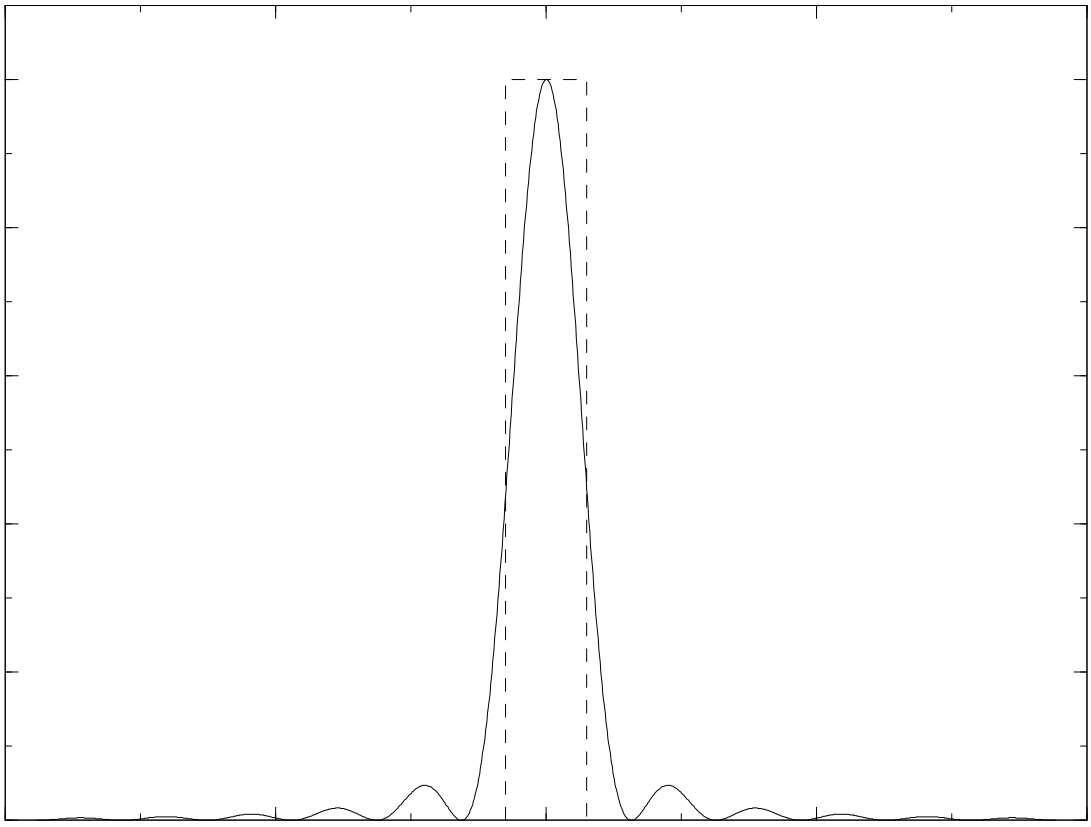

Figure 6.1: Power Spectrum of a Window Function (solid line) and of the Rectangular Filter with the same ENB (dashed line)

Effective Window Length: The application of a (non-rectangular) window function, puts more emphasis on the signal near the centre of an analysis window than near the edges. Hence, the length of the window is effectively reduced. There are several definitions of the effective length of a window, depending on the application. Unless specified differently, *xassp* uses the equal bandwidth criterion. That is: the effective length of a window is the length of a rectangular window whose transfer function has the same bandwidth. The reduction factor varies between about 1.3 for a triangular window and 2 for the most compact Blackman and Gaussian windows (see Harris (1978) for more details). The bandwidth definition used in this case is the Equivalent Noise Bandwidth.

Equivalent Noise Bandwidth (ENB): The Equivalent Noise Bandwidth (also called Equivalent Rectangular Bandwidth or ERB) of a window function equals the integral over frequency of the power spectrum of the window function, divided by the maximum level of the power spectrum. The term Equivalent Rectangular Bandwidth refers to the fact that this bandwidth is the same as the width of an (ideal) rectangular filter with the same peak power gain and the same integral

over frequency of its power spectrum (see Figure 6.1). The term Equivalent Noise Bandwidth refers to the fact that, if the window function were treated as a filter, it would output the same noise power as the (ideal) rectangular filter when both are excited by the same white noise (see Harris (1978) for more details).

Fast Fourier Transform (FFT): The FFT is a computer algorithm for calculating a DFT. It exploits symmetry properties in the calculation of the Fourier coefficients that occur when the number of coefficients (generally called the number of points) is a power of 2, to drastically reduce computation effort. This reduction is such, that even on modern computers and despite the rather strong restrictions on the number of points, the FFT is nearly always used to calculate a DFT.

Frame: Since the speech signal is by nature non-stationary, most analyses are short-term. This means that they process only a short signal stretch at a time. During this stretch (or frame as it is usually called) the signal is assumed to be quasi-stationary. Processing parameters involved are the frame size, the frame shift and, for most analyses, a window function and an effective window length.

Frame Shift: In a frame-based analysis, the analysis window is shifted along the signal in discrete, fixed, steps. In other words, the global signal features are analysed at a constant rate (the frame rate). In order not to leave parts of the signal unanalysed, a certain overlap should exist between adjacent frames. In general, an overlap of 50% or more is recommended, hence the default value of 10 ms for a frame size of 20 to 25 ms. Smaller values (larger overlap) will result in smoother contours at the cost of higher data rates. Larger values may result in data loss, especially for very compact windows like the higher-order Blackman and Gaussian.

As resource and in menus, frame shift is entered in milliseconds as the most convenient unit. Internally, however, it is set to the nearest integral number of samples. Therefore, the frame shift used may differ somewhat from the value you specified, especially for small values at low sampling frequencies.

Frame Size: The frame size or length of the analysis window determines the frequency resolution of the analysis. The larger the frame size, the better the spectral resolution. However, a large frame size will also result in *smearing* of fast transients such as plosives. In other words, the larger the frame size, the worse the temporal resolution.

For analysis of global features, such as LPC, formants or energy, a generally recommended compromise is a frame size of 20 to 25 ms. For very high-pitched voices a smaller frame size may be used to get a better temporal resolution. For very low-pitched voices a larger frame size may be necessary to reduce variation in the analysis results of voiced stretches due to the relative position of the frame with respect to the pitch period.

As resource and in menus, frame size is entered in milliseconds as the most convenient unit. Internally, however, it is set to the nearest integral number of samples. Therefore, the frame size used may differ somewhat from the value you specified, especially for small values at low sampling frequencies.

Gain: In sonagrams and sections the gain parameter is used to shift the display range upwards or downwards. A positive gain value shifts the sonagram more towards black and the spectrum in a section upwards. A negative value more to white and down, respectively. Note that gain does not influence the spectral levels themselves, but only which range of levels will be displayed. Gain is expressed in dB relative to an internally calculated estimate of the maximum spectral level. The default value of 0 dB should give a reasonable plot, provided the signal values span about the full dynamic range of the A/D convertor. The latter may be achieved by setting `autogain on` in your config.

Linear Predictive Coding (LPC): LPC has basically been developed for data compression in signal coding. In speech research, it is typically used for estimating the spectral envelope of the speech signal, which is then assumed to represent the transfer function of the vocal tract. Underlying LPC is the assumption that a sample value can be predicted on the basis of a linear combination of a number of previous sample values. In formula:

$$\hat{s}_n = \sum_{m=1}^{M} a_m s_{n-m}, \qquad \text{with } M \text{ the prediction order}$$

Since, in the analysis, the sample value s_n is known, the prediction error

$$e_n = |s_n - \hat{s}_n|$$

can be calculated for the set of prediction coefficients $a_1 \ldots a_M$. For each frame, the LPC analysis determines the prediction coefficients in such a way that the energy of the error signal within that frame is minimal. The prediction coefficients then form the coefficients of a digital filter that optimally *flattens* the signal spectrum. In other words, it will have anti-resonances where the signal spectrum has resonances. The inverse of this filter can therefore be used to estimate the spectral envelope of the signal. The LPC analysis method used in *xassp* is called the autocorrelation method and uses the Durbin recursion to determine the prediction coefficients on the basis of autocorrelation coefficients (for more details, see e.g. Makhoul 1975 or Markel and Gray 1976).

LPC Order: The LPC order (or number of coefficients) determines how closely the inverse LPC spectrum models that of the signal. Since, in speech analysis, we are mainly interested in the spectral envelope, the optimal order relates to the length of the vocal tract of the speaker and the sampling frequency used (see Markel and Gray 1976). The rule of the thumb is: sampling frequency in kHz + 2 or 3 for male voices and about 10% fewer for female voices. Thus, for a sampling frequency of 16 kHz: 18 or 19 for a male voice and 16 to 17 for a female voice. Note that two coefficients are needed to model one resonance (formant). Therefore, in the above example for a male voice, maximally 9 formants can be modelled, which corresponds to the rule of the thumb: 1 formant per kHz bandwidth. The extra formant follows from a potential nasal formant or the coefficients may be used to compensate for an anti-resonance. The odd coefficient may be included to model a very global aspect of the spectrum. If, for example, no pre-emphasis is applied prior to the analysis, this coefficient can model the overall spectral tilt.

If the LPC order is chosen too low, only the strongest resonances will be modelled and may undergo a frequency shift. If the order is chosen too high, strong harmonics will be treated as resonances.

Pre-emphasis: Pre-emphasis is used to change the overall spectral tilt of the speech signal. It implies a first-order filter with the transfer function:

$$H(z) = 1 + uz^{-1}, \quad \text{with} \quad -1 \leqslant u \leqslant 1.$$

A positive value of u amplifies the lower frequencies, while a negative value amplifies the higher frequencies. Typically, u is chosen near -1 (e.g. -0.95) to compensate the -6 dB/oct slope of the spectrum of voiced speech. Such a compensation is necessary in a sonagram in order to show the higher formants. Although a positive value should be used for unvoiced speech (which usually has a positive spectral tilt), the fixed negative value compensates for the lower amplitudes of such stretches. Note that some signal processing packages use a definition with a different sign for u.

Range: The range parameter sets the difference between the highest and the lowest spectral level to be displayed in a sonagram or a section. For a sonagram, the default value is 50 dB. Choosing a smaller value may result in components of interest becoming invisible. Choosing a higher value will make the picture less distinct, due to the limited resolving power of grey levels. Moreover, too high a range may lead to sidelobes due to the window function becoming visible. For a Section, the range may be set larger because the visual resolution is far better and sidelobes are easier to recognise as such. The default value here is 80 dB, also because no pre-emphasis is applied.

Window Function: In a frame-based analysis, the signal is implicitly assumed to be zero outside the frame. This is equivalent to multiplying the signal with a window function which equals 1 within the frame and 0 outside it. The *sharp* edges of this function cause distortions of the signal spectrum. The two main effects are:

- a broadening of the frequency components (width of the main lobe)
- the occurrence of spurious components (sidelobes)

The second effect is the most serious one, because, for a rectangular window, the maximum level of the sidelobes is only 13 dB below that of the main lobe. Therefore, the sidelobes of a single strong signal component may mask other components completely or suggest energy at frequencies where there are in fact no signal components. In order to reduce these sidelobes, a *tapered* window function (a function that gradually approaches 0 at its ends) is usually applied prior to the analysis. For any window function there is a trade-off between the width of the main lobe and the maximum level of the sidelobes: The lower the relative level of the sidelobes, the wider the main lobe. In sonagrams and sections, the resulting increase in the analysis bandwidth is compensated for. For more details on window functions and their figures of merit, see Harris (1978). In *xassp*, a large number of window functions are available. These include the *classical* triangular (Bartlett), hanning (or Hann) and Hamming windows, and two groups, viz. the Blackman and Gaussian windows. For most analyses, the Hamming window with a highest sidelobe level of -43 dB is a good choice. For the Sonagram, the minimum 3-term Blackman-Harris window has been chosen because of its even lower sidelobe level of -67 dB.

References

Barry, W. J., W. v. Dommelen, H. Janßen, K. J. Kohler, K. Schäfer, W. Thon, and G. Timmermann (1982). *Phonetic Data Processing at Kiel University*. AIPUK 18.

Carlson, R. and B. Granström (1986). A search for durational rules in a real-speech data base. *Phonetica 43*, 140–154.

Harris, F. J. (1978). On the use of windows for harmonic analysis with the discrete fourier transform. In *Proceedings of the IEEE*, Volume 66, pp. 51–83. IEEE.

IPDS (1994). *The Kiel Corpus of Read Speech*, Volume 1, CD-ROM#1. Kiel: Institut für Phonetik und digitale Sprachverarbeitung.

IPDS (1995). *The Kiel Corpus of Spontaneous Speech*, Volume 1, CD-ROM#2. Kiel: Institut für Phonetik und digitale Sprachverarbeitung.

IPDS (1996). *The Kiel Corpus of Spontaneous Speech*, Volume 2, CD-ROM#3. Kiel: Institut für Phonetik und digitale Sprachverarbeitung.

IPDS (1997). *The Kiel Corpus of Spontaneous Speech*, Volume 3, CD-ROM#4. Kiel: Institut für Phonetik und digitale Sprachverarbeitung.

Kohler, K. J., M. Pätzold, and A. P. Simpson (1995). *From scenario to segment: the controlled elicitation, transcription, segmentation and labelling of spontaneous speech*. AIPUK 29.

Ladefoged, P. (1996). *Elements of Acoustic Phonetics*. Chicago: The University of Chicago Press.

Makhoul, J. (1975). Linear prediction: A tutorial review. In *Proceedings of the IEEE*, Volume 63, pp. 561–580. IEEE.

Markel, J. D. and A. H. Gray (1976). *Linear Prediction of Speech*. Berlin/Hamburg/New York: Springer.

Rosen, S. and P. Howell (1991). *Signals and systems for speech and hearing*. London/San Diego: Academic Press.

Schäfer-Vincent, K. (1982). Significant points: Pitch period detection as a problem of segmentation. *Phonetica 39*, 241–253.

Schäfer-Vincent, K. (1983). Pitch period detection and chaining: Method and evaluation. *Phonetica 40*, 177–202.

Scheffers, M. and W. Thon (1991). Workstation and signal processing software for experimental phonetics. In *Proceedings of the XIIth International Congress of Phonetic Sciences*, Volume 2, Aix-en-Provence, France, pp. 486–489. Université de Provence.

Wells, J. C., W. J. Barry, and A. J. Fourcin (1989). Transcription, labelling and reference. In A. J. Fourcin, G. Harland, W. J. Barry, and V. Hazan (Eds.), *Speech Technology Assessment. Towards Standards and Methods for the EUROPEAN COMMUNITY*, pp. 141–159. Chichester: Ellis Horwood.

Appendix A

Key and Mouse Bindings

Key	Action
<**Shift**>**LM**	set temporary marker
F3	move temporary marker to next zero crossing on the left
F4	move temporary marker to next positive zero crossing on the right
F5	set left bracket on zero crossing[a]
F6	set right bracket on zero crossing[a]
<**Ctrl**>**F5**	set left bracket onto cursor position[b]
<**Ctrl**>**F6**	set right bracket onto cursor position[b]
<**Shift**>**F5**	clear left bracket
<**Shift**>**F6**	clear right bracket
Return	clear brackets[c]
Del	clear brackets
Esc	close window
Space	play signal from start to end or stop playing
Backspace	clear temporary marker
`w`	log data to file (only when data logging is enabled)
<**Ctrl**>`w`	log data to file and add a comment line

[a] The brackets are set onto the next zero crossing to the left of the cursor only if the resource `setOnZeroCrossing` (see Section B.4) is set to `True`. If the resource is `False`, the brackets are set onto the cursor position.

[b] This key sets the bracket onto the cursor position and does not care about the `setOnZeroCrossing` resource.

[c] **Return** does not work on all machines

Table A.1: Commonly Used Key Bindings in *xassp*

Mouse button	Action
LM	play from (temporary marker/left bracket/beginning of window)[a] to cursor
MM	play from (temporary marker/left bracket/beginning of window)[a] to (right bracket/end of window)[a]
RM	play from cursor to (right bracket/end of window)[a]
<**Alt**>**RM**	post window menu
<**Ctrl**>**RM**	post edit menu

[a] *xassp* takes the first item from the slash-separated lists that is present when determining what to play. If a temporary marker is present and you press the left mouse button, this marker determines the beginning of the region to be played. If the temporary marker is not present, but the left bracket is, then the region starts at the position of the left bracket. If none of these are currently displayed, the playing starts at the beginning of the window.

Table A.2: Commonly Used Mouse Bindings in *xassp*

Key	Action
<**Shift**>s	open a new section window
CursorUp	increase the vertical resolution of the speech signal
CursorDown	decrease the vertical resolution of the speech signal
Home	restore the original vertical resolution of the speech signal

Table A.3: Key Bindings in the Speech Signal Window

Key	Action
f	toggles the updating of the section when cursor in speech signal window is moved (same as the *Freeze/Follow* menu item)
k	keeps the current section (same as the *Keep* menu item)
c	removes the last kept section (same as the *Clear* menu item)

Table A.4: Key Bindings in the Section Window

Key/Mouse button	Action
<**Shift**>**LM**	set temporary marker
<**Shift**>**MM**	move label unchanged to the point in time defined by the temporary marker
<**Ctrl**>**LM**	move label with uncertainty marker %
<**Alt**>**LM**	move label and mark as deleted (appended -))
<**Alt**>**MM**	move label and edit name
<**Ctrl**>**MM**	insert a label at the temporary marker
CursorLeft	move the temporary marker to the next label on the left
CursorRight	move the temporary marker to the next label on the right
h	insert aspiration or creak label (after a plosive or glottal stop, respectively)
z	insert a lengthening label
<**Ctrl**>s	opens the *Go to* dialog box (same as selecting the menu item *Go to*)
<**Ctrl**>f	jumps to the next occurrence of the label that was last searched for
<**Ctrl**>b	jumps to the previous occurrence of the label that was last searched for
<**Ctrl**>l	jumps to the last label that was set

Table A.5: Key and Mouse Bindings in the Label Window

Appendix B

X Resources

B.1 Resources for the *Core* widget

The *Core* widget is defined by the *Intrinsics* toolkit. Because many of its resources are seldom needed by the user, only the most frequently used ones will be listed. For a complete list see the *Core* manual page, which is normally included in common Motif distributions.

background: Specifies the background colour for the widget.

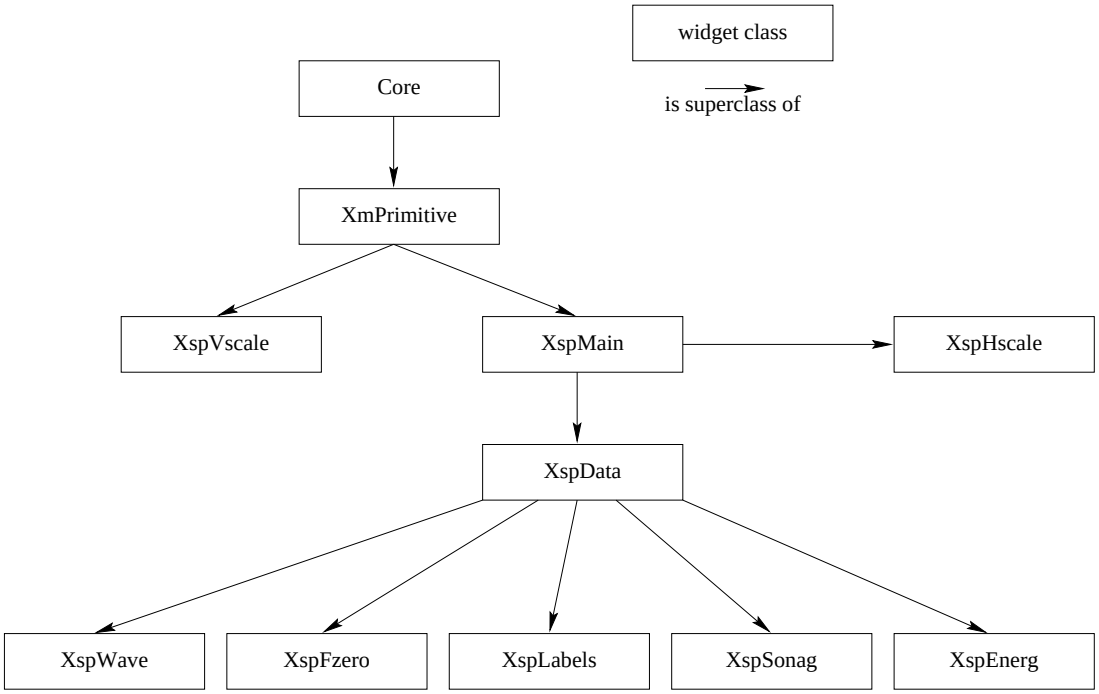

Figure B.1: *xassp* widget hierarchy

backgroundPixmap: Specifies a pixmap for tiling the background. The first tile is placed at the upper left corner of the widget's window.

borderColor: Specifies the colour of the border in a pixel value.

borderPixmap: Specifies a pixmap to be used for tiling the border. The first tile is placed at the upper left corner of the border.

borderWidth: Specifies the width of the border that surrounds the widget's window on all four sides. The width is specified in pixels. A width of 0 (zero) means that no border shows.

height: Specifies the inside height (excluding the border) of the widget's window.

translations: Points to a translations list. A translations list is a list of events and actions that are to be performed when the events occur.

width: Specifies the inside width (excluding the border) of the widget's window.

x: Specifies the x-coordinate of the upper left outside corner of the widget's window. The value is relative to the upper left inside corner of the parent window.

y: Specifies the y-coordinate of the upper left outside corner of the widget's window. The value is relative to the upper left inside corner of the parent window.

B.2 Resources for the *XmPrimitive* widget

The *XmPrimitive* widget class is defined by the *Motif* toolkit. Since many of its resources are seldom used, only the most important ones are listed. For a complete list see the *XmPrimitive* manual page.

bottomShadowColor: Specifies the colour to use to draw the bottom and right sides of the border shadow.

bottomShadowPixmap: Specifies the pixmap to use to draw the bottom and right sides of the border shadow.

foreground: Specifies the foreground drawing colour used by Primitive widgets.

highlightColor: Specifies the colour of the highlighting rectangle.

highlightOnEnter: Specifies whether the highlighting rectangle is drawn when the cursor moves into the widget. The default is False.

highlightPixmap: Specifies the pixmap used to draw the highlighting rectangle.

highlightThickness: Specifies the thickness of the highlighting rectangle.

shadowThickness: Specifies the size of the drawn border shadow.

topShadowColor: Specifies the colour to use to draw the top and left sides of the border shadow.

topShadowPixmap: Specifies the pixmap to use to draw the top and left sides of the border shadow.

B.3 Resources for the *XspMain* widget

cursorColor: Specifies the cursor colour.

dataColor: Specifies the color that is used to draw the data.

dataMargin: Specifies the amount of empty space that is added to the left and the right of the displayed data.

gridColor: Specifies the grid colour.

gridLength: Specifies the length of a single grid line.

gridSpace: Specifies the horizontal space between grid lines

leftBracketColor: Specifies the the colour of the left bracket.

maximum: Specifies the maximum data value for display.

minimum: Specifies the minimum data value for display.

pixPerSpl: Specifies the resolution used to display the data in pixels per millisecond.

rightBracketColor: Specifies the colour of the right bracket.

showCursor: Specifies whether a cursor should be drawn.

showGrid: Specifies whether to draw a horizontal grid.

splPerSec: Specifies the sampling frequency in samples per second.

tempMarkColor: Specifies the colour of the temporary marker.

tempMarkLineStyle: Specifies the line style of the temporary marker. This can be solid, onoffdash, or doubledash.

B.4 Resources for the *XspData* widget

labelLinesColor: Specifies the colour of the label lines.

scrollbarIncrement: Specifies the fraction of the window width that the data should be scrolled if the scroll bar arrows are used.

scrollbarPageIncrement: Specifies the fraction of the window width that the data should be scrolled if the user clicks inside the scroll bar.

setOnZeroCrossing: Specifies whether the brackets should automatically be set on the next zero crossing to the left of the cursor.

showLabelLines: Specifies whether label lines from a linked label window should be displayed.

valueUnit: Specifies the value to be used in the status line value label.

B.5 Resources for the *XspWave* widget

The *XspWave* widget class defines no new resources.

B.6 Resources for the *XspFzero* widget

drawMode: Specifies how to draw the data values. This can be lines, circles, or rectangles.

frameShift: Specifies the shift of the analysis window.

logarithmic: Specifies whether to use a logarithmic vertical scale.

maxFzero: Specifies the highest F0 value to be analyzed.

minFzero: Specifies the lowest F0 value to be analyzed.

noiseAmp: Specifies the minimum speech signal amplitude considered relevant for analysis.

B.7 Resources for the *XspLabels* widget

bottomMargin: Specifies the space in pixels between window bottom and labels.

highlightColor: Specifies the colour to highlight labels with.

horizontalSpacing: Specifies the space in pixels to be left between the label and the vertical label line.

labelFont: Specifies the label font.

truncate: Specifies whether labels that do not fit into the window are truncated, or other labels are overwritten if there is no space left.

verticalSpacing: Specifies the vertical space in pixels between stacked labels.

B.8 Resources for the *XspSonag* widget

bandwidth: Specifies the bandwidth of the DFT spectrum.

gain: Specifies the gain in dB.

lpcLength: Specifies the effective length of the LPC analysis window.

lpcMode: Specifies whether to compute an LPC (True) or DFT (False) spectrum.

lpcOrder: Specifies the order of the LPC analysis.

numFFT: Specifies the number of FFT points.

numLevels: Specifies the number of grey levels to be used.

preEmphasis: Specifies the pre-emphasis.

range: Specifies the range in dB.

shift: Specifies the shift of the analysis window.

wfunction: Specifies the analysis window function.

B.9 Resources for the *XspSection* widget

average: Specifies whether to compute an average spectrum.

avrLength: Specifies the length of the signal part the average spectrum should be computed from.

bandwidth: Specifies the bandwidth of the DFT spectrum.

gain: Specifies the gain in dB.

listFormants: Specifies whether formants should be analyzed and displayed

lpcLength: Specifies the effective length of the LPC analysis window.

lpcMode: Specifies whether to compute an LPC (True) or DFT (False) spectrum.

lpcOrder: Specifies the order of the LPC analysis.

numFFT: Specifies the number of FFT points.

numFormants: Specifies the number of formants to analyze

preEmphasis: Specifies the pre-emphasis.

range: Specifies the range in dB.

sectionColors: This is a comma-separated list of colours that are used if more than one section is displayed.

wfunction: Specifies the analysis window function.

B.10 Resources for the *XspEnerg* widget

drawMode: Specifies how to draw the data values. This can be lines, circles, or rectangles.

frameShift: Specifies the frame shift.

frameSize: Specifies the frame size.

wfunction: Specifies the window function.

B.11 Resources for the *XspHscale* widget

labelFactor: Specifies a factor for multiplication of the scale labels.

labelFont: Specifies the font to be used for the tick labels.

labelMargin: Specifies the margin between tick labels and tick marks.

labelUnit: Specifies a unit that is appended to each label.

relTicLength: Specifies the length of the shorter tick marks in percent of the length of the longer ones.

ticHeight: Specifies the length of the longer tick marks.

B.12 Resources for the *XspVscale* widget

factor: Specifies the factor for multiplication of each label.

labelColor: Specifies the colour of the scale labels.

labelFont: Specifies the scale label font.

labelMargin: Specifies the margin between labels and tick marks.

precision: Specifies the number of digits to appear after the decimal-point.

ticColor: Specifies the colour of the scale tick marks.

unit: Specifies the unit to be appended to each scale label.